GHAJAR
EXHIBIT 58

UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA

RICHARD KADREY, an individual, et al. v. META PLATFORMS, INC., a Delaware corporation; Defendant.	Case No. 3:23-cv-03417-VC

REBUTTAL EXPERT REPORT OF BARBARA FREDERIKSEN-CROSS

Signed in Hubbard, Oregon on February 10, 2025

[illegible]

Barbara Frederiksen-Cross

Table of Contents

I. ASSIGNMENT 3

II. QUALIFICATIONS 3

III. SUMMARY OF OPINIONS 5

IV. BACKGROUND ON BITTORRENT AND TORRENTS 8

A. BASICS OF ACCESSING FILES OVER THE INTERNET 10

B. BITTORRENT 19

C. TORRENT PROCESS 22

D. USE CASES OF BITTORRENT PROTOCOL 26

V. OVERVIEW OF THE RELEVANT SOURCE CODE AND AT-ISSUE DATASETS 28

A. THE KREIN REPORT MISREPRESENTS META'S TORRENT SOURCE CODE 28

B. THE KREIN REPORT MISREPRESENTS THE DATASETS DOWNLOADED BY META 31

VI. IT IS HIGHLY UNLIKELY THAT META SEEDED PLAINTIFFS' WORKS 34

A. SEEDING OF A TORRENT CAN ONLY OCCUR AFTER THE DOWNLOAD OF A TORRENT IS COMPLETE 35

B. META'S IMPLEMENTED SAFEGUARDS RENDERED THE POSSIBILITY OF SEEDING HIGHLY UNLIKELY 36

C. SEEDING PLAINTIFFS' WORKS IS ALSO UNLIKELY DUE TO META'S NETWORK CONFIGURATION AND OTHER FACTORS 42

VII. CONCLUSION 57

VIII. APPENDIX A: VALIDATION OF META'S NETWORK CONFIGURATION 59

IX. APPENDIX B: METHODOLOGY FOR COMPUTING SIZE OF PLAINTIFFS' WORKS WITHIN AT-ISSUE DATASETS 62

X. APPENDIX C: DATA TABLES FOR SIZES OF PLAINTIFFS' WORKS ACROSS AT-ISSUE DATASETS 67

A. FILE-LEVEL ANALYSIS OF SIZES OF PLAINTIFFS' WORKS 67

B. PIECE-LEVEL ANALYSIS OF SIZES OF PLAINTIFFS' WORKS 72

XI. APPENDIX D: MATERIALS CONSIDERED 78

XII. APPENDIX E: CURRICULUM VITAE 83

I. ASSIGNMENT

1. I have been retained by counsel for Meta Platforms Inc. ("Meta") to provide expert analysis in the matter *Kadrey et al v. Meta Platforms Inc.*, Case Number: 3:23-cv-03417-VC. I have been asked to prepare an expert report in response to aspects of the Opening Expert Report of Jonathan Krein, dated January 10, 2025 ("Krein Report") regarding the BitTorrent protocol, and allegations that Meta "seeded" Plaintiffs' works.

2. I am informed by counsel that, in order to show infringement of their copyrighted works, Plaintiffs must show that they own a valid copyright in their books and that Meta copied protectable elements of those books. I am further informed by counsel that Plaintiffs bear the burden of proving that Meta infringed their copyrights. I am further informed that, in connection with the use of the BitTorrent protocol, Plaintiffs allege that Meta distributed Plaintiffs' works to third parties by "seeding" those works.

3. The opinions in this report are based on my research and experience in the field of computer software and computer networking, specifically the BitTorrent protocol and clients such as "libtorrent," as well as the materials I reviewed in preparation of this report which are listed in **Appendix D**. Support for the development of this report was provided by Keystone Strategy LLC, and by my colleagues at JurisLogic, working under my direction and supervision. The opinions expressed here are mine alone.

II. QUALIFICATIONS

4. I am the Director of Litigation Services for JurisLogic, LLC ("JurisLogic"). JurisLogic is an Oregon corporation that provides consulting services to computer hardware and software manufacturers and computer-related technical assistance to the legal profession in the United States, Canada, Japan, China, Europe, and the UAE. My experience includes software design, programming, project management, capacity planning, performance tuning, problem diagnosis, and administration of hardware, operating systems, application software, and database management systems. I have over fifty years of personal experience as a software developer and consultant, including the analysis of computer-based data, software development of web-based systems, and software development for secure online data systems used by banks, insurance companies, hospitals, and telecommunication providers.

5. I have experience in the design, development, and analysis of computer software, and I have previously provided both trial and deposition testimony as an expert for matters in federal and state courts, authored a number of papers, and delivered lectures on technology to the legal profession. My curriculum vitae is attached as **Appendix E** to this report and lists the publications I have authored in the last 10 years and the cases in which I have testified by deposition or at trial in the last 4 years.[1]

6. I also have prior experience with the analysis of peer-to-peer ("P2P") file distribution networks, including both the analysis of BitTorrent-related source code and testing related to BitTorrent's operation. This experience includes tests in which I created torrent files, used BitTorrent to upload and download files, used BitTorrent to transfer files over computer networks, and captured and analyzed BitTorrent network traffic. I have also studied both publicly available BitTorrent source code (including libtorrent[2] and other implementations), as well as source code for proprietary file distribution software that uses the BitTorrent protocol, and proprietary systems used to monitor BitTorrent activity.

7. My background and experience also include the design, implementation, and ongoing administration of databases and multi-dimensional data aggregation systems, including data extraction, transfer, and load operations used for data analysis platforms. My experience also includes programming for embedded and robotic systems. I also have experience with computer and network capacity management, storage management, and disaster recovery planning and testing.

8. I have previously qualified as an expert in federal and state courts to testify about the operation of computer software and computer systems, including for matters that involve software development disputes, failed software systems, and patent, copyright and trade secret disputes. I have also previously testified as an expert in several litigation matters that involved BitTorrent technologies. My work in these other matters included analysis and testimony

[1] **Appendix E**, Curriculum Vitae of B. Frederiksen.

[2] "Libtorrent," libtorrent, accessed February 5, 2025, https://www.libtorrent.org/.

relating to systems that monitor P2P file distribution systems and send notices to Internet Service Providers ("ISPs") based on the activity they detect.

9. I am a member of the Institute of Electrical and Electronics Engineers ("IEEE") and the Association for Computing Machinery ("ACM").

10. I am a salaried employee of JurisLogic, which specializes in providing consulting services to corporations and attorneys on intellectual property matters (such as copyright and patent infringement matters, and misappropriation of trade secrets) and performing assessments of computer software, large scale computer data productions, software-controlled devices, and software development projects. I am also one of the principals of JurisLogic. JurisLogic is compensated at the rate of $595 an hour for my work in this case. My compensation as a principal of JurisLogic, and any compensation that JurisLogic receives from my services on this case, does not in any way depend on the substance of my opinions or the outcome of this or any other case.

11. If called upon to testify at trial, I may present oral testimony and/or tutorials about the operation of BitTorrent, the evidence I analyzed, my analysis processes, and the opinions I formed based on my analysis.

12. In addition, I understand that I may testify regarding my opinions on related matters, including those raised at trial by Plaintiffs' attorneys and experts, or the Court, concerning these issues. I reserve the right to supplement my report in the event that new facts become known to me before trial that impact my opinions or the bases therefor and to respond to responses to my opinions. I am aware of the continuing obligation to supplement my report under Rule 26 of the Federal Rules of Civil Procedure.

III. SUMMARY OF OPINIONS

13. In this report, I respond to several assertions the Krein Report makes regarding Meta's use of "BitTorrent," the alleged "seeding" of data, and Meta's environment for performing torrent downloads. Based on my analysis, I demonstrate that the Krein Report (i) misunderstands and mischaracterizes the BitTorrent protocol when suggesting that Meta seeded data, (ii) does not consider the safeguards implemented by Meta to prevent the seeding of downloaded data, nor

the myriad of unlikely factors that would have needed to occur for seeding to take place, and (iii) presents no evidence that Plaintiffs' (or any) works were actually seeded by Meta, and rather only speculates that this was the case based on the existence of torrenting scripts in Meta's codebase.

14. First, based on its static and isolated analysis of Meta's source code files,[3] the Krein Report presents conjecture on the possibility of additional datasets potentially torrented by Meta. In **Section V.B** below, I present evidence that the Krein Report misstates the datasets torrented by Meta.

15. Then, in its description of the "BitTorrent" protocol, the Krein Report misrepresents fundamental concepts of the protocol in claiming that Meta necessarily seeded the at-issue data. The Krein Report inaccurately asserts that (i) "[a]ny computer using a BitTorrent client/library application becomes a host for any data they download," (ii) "[t]his reciprocal condition of hosting is called "seeding," and (iii) "when Meta torrented LibGen, the LibGen data was seeded back out to other peers in the network.""[4] This description mischaracterizes the core components of the BitTorrent protocol as employed by Meta, ignores the safeguards implemented by Meta to prevent data seeding, and ignores the confluence of several technical factors that would have needed to occur for Meta to have seeded Plaintiffs' works to others on the BitTorrent network. As I describe below, these ignored factors make it highly unlikely that Meta seeded Plaintiffs' works.

16. With respect to seeding (which refers to the ability of a peer on the BitTorrent, after the download of the torrented data is complete, to share the data with other peers on the network),[5] as I discuss in **Section IV.B**, Meta had safeguards built in that prevent seeding such as (i) removing the torrent from the session within no more than 60 seconds after the torrent download has completed, and (ii) blocking all unsolicited inbound connections, effectively rejecting requests from un-connected leechers during the maximum 60 second period.

---

[3] Krein Report ¶¶127, 168.

[4] Krein Report ¶¶118, 127.

[5] Krein Report ¶119; See also: BitTorrent Limited, "Help Center - What Is Seeding?," BitTorrent, accessed February 8, 2025, https://www.bittorrent.com/en/support/solutions/articles/29000041669-what-is-seeding-.

17. Further, in order for Meta to have seeded Plaintiffs' works, many different practical and technical factors would have had to come together that, in my opinion, make it highly unlikely that this could have occurred. Specifically, **<u>all</u>** of the following must have been true:

    a. Meta must have initiated the connection with the particular leecher in question. As I discuss in **Section VI.C.1**, Meta's network configurations would have blocked any connections not initiated by Meta,

    b. The leecher must not already have possessed the particular pieces of the torrent containing Plaintiffs' works. If the leecher already possesses these pieces then it will not require them from Meta,

    c. The leecher must have prioritized the pieces containing Plaintiffs' works (which represent small portions of the datasets downloaded using BitTorrent). If the pieces are not prioritized, Meta's code may remove the torrent (within the maximum of 60 seconds), before the pieces can be seeded to the leecher,

    d. The leecher must have randomly selected Meta as the seeder among available peers from which these specific pieces are to be obtained. The leecher would be equally likely to receive the pieces containing the Plaintiffs' works from any of the other randomly selected peers that had those pieces,

    e. The leecher must have been able to receive the pieces from Meta before the torrent was removed from the session (within a maximum of 60 seconds),

    f. Finally, a leecher must have remained in Meta's unchoked slots long enough to download a piece containing Plaintiffs' works.

18. I note that the Krein Report does not point to any evidence that any one of these preconditions to seeding Plaintiffs' works was present, much less all of them. Nor does the Krein Report show that Meta actually seeded any data, let alone any of Plaintiffs' works, and any suggestion that this occurred is pure conjecture.

19. In sum, based on the safeguards implemented by Meta, the numerous factors in the torrent network that would have needed to align, and the de minimus proportion of Plaintiffs' works to the at-issue datasets (as discussed in **Section VI.C.2**), it is highly unlikely that Meta seeded the Plaintiffs' works to other peers.

20. I begin with a description of the requisite technologies involved in downloading files over the internet and specifically using the BitTorrent protocol in the following section.

IV. BACKGROUND ON BITTORRENT AND TORRENTS

21. The Krein Report asserts that "Meta directly downloaded at least one copy of LibGen and torrented at least one copy of LibGen via BitTorrent methods,"[6] further stating that certain datasets were "seeded."[7] To respond to these assertions, it is first necessary to provide a background and understanding of the different technologies that underlie the torrenting process. In the following section, I first describe the basic components involved in accessing files over networks, followed by a description of peer-to-peer networking. Then, in **Section IV.B** below, I detail the BitTorrent protocol and introduce key terms. In **Section IV.C,** I discuss the process of downloading a file using the BitTorrent process. Finally, in **Section IV.D** I outline a few contemporary uses of the protocol.

22. The cornerstone of modern computing is the ability to access information almost instantaneously. The advent of technologies enabling such access began out of the need to efficiently access information among scientists and governments spread across the world. First designed to overcome this problem, ***computer networks*** are a set of interconnected computers with the ability to communicate information via wired or wireless methods.[8] The ***internet*** is composed of multiple networks that are all interconnected to facilitate the efficient flow of information.[9]

---

[6] The Krein Report, ¶28.

[7] The Krein Report, §§ 9.3.4, 10.2.4.

[8] "What Is Computer Networking? | IBM," July 1, 2024, https://www.ibm.com/think/topics/networking.

[9] Computer Networking: A Top-Down Approach, p. 28.

23. To communicate via the internet, computers utilize standardized ***protocols*** that represent a set of rules and instructions that determine how information is transmitted between different entities.[10] For instance, on the internet, computers utilize the ***Internet Protocol*** (IP) to send data, assembled in a standardized format that can be understood by both the sender and the recipient of the data.[11] As another example, most websites are accessed using the ***Hypertext Transfer Protocol*** (HTTP),[12] which was initially designed and commonly used in the 1990s for communicating between a ***web browser*** and a ***web server***,[13] but has evolved to be used in many applications such as machine-to-machine communication or programmatic access to other programs.[14] A web browser is a software program that enables users to view and interact with content by interacting with a web server, which is a computer that possesses the requisite content on the internet.

24. The web browser relies on a ***Uniform Resource Locater*** (URL) that points to a specific resource on the internet and is itself comprised of various parts such as the data needed to identify (i) protocol used for communication, (ii) a ***domain name*** which is the name of the website requested to be accessed by the user,[15] and (iii) a ***path*** within the domain that would contain the required information or service.[16] In practice, a user wishing to access content over the internet utilizes a web browser and inputs a URL, which instructs the browser to locate the content on a web server and access the requisite information or service. In the next section, I further explain the technical nuances that underlie the process of accessing large files over the internet.

[10] Computer Networking: A Top-Down Approach, p. 35.

[11] Computer Networking: A Top-Down Approach, p. 31.

[12] Computer Networking: A Top-Down Approach, p. 129.

[13] Computer Networking: A Top-Down Approach, p. 130.

[14] A popular implementation is in the form of Application Programming Interface (API) that run on top of HTTP to exchange information between multiple applications. See: Computer Networking: A Top-Down Approach, p. 468.

[15] As discussed below, the domain name is further converted into a unique identifier for a server where the information would be accessible, using a Domain Name System (DNS). See: Computer Networking: A Top-Down Approach, p. 160.

[16] Computer Networking: A Top-Down Approach, p. 129-130.

A. Basics of Accessing Files over the Internet

25. As explained above, the internet allows for and facilitates access to information over interconnected computer networks, communicated from a computer containing the requisite information to another computer requesting it. In this section, I outline the processes involved in accessing files and data over the internet, as well as different protocols that enable this flow of information. I also introduce peer-to-peer networks, which present unique advantages for utilizing the resources available on the network.

26. For every network, there is a finite speed that data that can be transmitted from one location to the other; this is referred to as the ***bandwidth*** of the network.[17] More specifically, the bandwidth of a network is defined by the maximum amount of data that can be transmitted in a given time (usually measured in bits[18] per second (bps)).

27. Bandwidth is a fundamental constraint imposed on the network due to physical limitations of the network infrastructure like cables, routers, and wireless frequencies used.[19] Over time, advancements in networking technologies, such as use of fiber optic cables in place of copper cables,[20] have significantly increased data transfer rates. Still, inherent limitations in bandwidth necessitate careful consideration to ensure data is transmitted reliably and efficiently.

28. An advancement in networking that revolutionized the methods of transmitting data with limited bandwidth in the 1960s, was that of ***packet-switching***.[21] With packet switching, any information transmitted over a vast network is first broken down into small units of data,

[17] "Network Bandwidth: What Is Bandwidth in Networking? - IT Glossary | SolarWinds," accessed February 5, 2025, https://www.solarwinds.com/resources/it-glossary/network-bandwidth.

[18] A bit is the smallest unit of data that can be processed by computers, represented by one of two values: 1 or 0. See: "Definition of Binary Digit (Bit) - Gartner Information Technology Glossary," Gartner, accessed February 5, 2025, https://www.gartner.com/en/information-technology/glossary/bit-binary-digit.

[19] "What Is Network Bandwidth and How Is It Measured?," Search Networking, accessed February 5, 2025, https://www.techtarget.com/searchnetworking/definition/bandwidth.

[20] Charter Communications, "Why Is the Bandwidth of Optical Fiber High? | Spectrum Enterprise," accessed February 5, 2025, https://enterprise.spectrum.com/content/spectrum/enterprise/en/home/support/faq/internet/why-is-the-bandwidth-of-optical-fiber-high.html.

[21] "Packet Switching," ETHW, February 17, 2024, https://ethw.org/Packet_Switching.

referred to as ***packets***.[22] Each individual packet is then sent over the network from the sender's computer to the recipient's computers using ***routers***, which analyze the packet's destination address and decide the best path to forward it such that it is ultimately received by the appropriate recipient.[23,24] Thus, the path from the sender to the recipient for a packet is determined dynamically, allowing for packets to take alternative paths to reach the recipient over a network, instead of following a pre-determined path.[25] In this way, packet-switched networks alleviate some concerns around limited bandwidth by breaking larger data transmissions into smaller pieces of information, and choosing the best network to transmit them so as to avoid congestion due to limited bandwidth.

29. As discussed, the Internet Protocol (IP) is used to transmit each packet to its destination. However, in addition to IP, another related protocol is used in conjunction to ensure that the information is received accurately, i.e., in the proper order and error-free. This protocol is referred to as the ***Transmission Control Protocol*** (TCP).[26] Together, the TCP/IP protocol suite is the foundation of many interactions over the internet, encompassing multiple layers and protocols that enable communication and data exchange.[27]

30. Packet switched networks were devised as a way to ensure reliability of communication sent over physical networks, which is evident given that packet switching provided a way to communicate information even in case of damaged or unreliable communication links along a given path. The utilization of packet switching has enabled a host of improvements for sending large amounts of data over a network, including reliability of communication, efficiency of transmission, and scalability of the network. The same concept of breaking large amounts of data into smaller units to boost reliability, efficiency, and scalability of the network is also

[22] Computer Networking: A Top-Down Approach, p. 50.

[23] "What Is a Router? | Router Definition," accessed February 5, 2025, https://www.cloudflare.com/learning/network-layer/what-is-a-router/.

[24] Computer Networking: A Top-Down Approach, p. 53.

[25] Computer Networking: A Top-Down Approach, p. 358.

[26] Computer Networking: A Top-Down Approach, p. 31.

[27] Computer Networking: A Top-Down Approach, p. 31.

applied to peer-to-peer (P2P) networks such as BitTorrent as I explain further in the **Section IV.A.3** below.

1) IP Addresses

31. Today, access to the internet is provided by ***Internet Service Providers*** (ISPs), organizations that manage the wired and wireless connections that power computer networks.[28] ISPs build and maintain a vast network infrastructure, including routers and fiber optic cables, which in turn form the backbone of the internet. In addition to the critical role of providing the necessary infrastructure, ISPs also facilitate the registration of domain names, and the assignment of unique addresses for each computer connected to the internet otherwise known as ***IP addresses***.[29]

32. An IP address is a series of numbers that act as unique identifiers for the computer connecting to the internet and its location on the internet, at a point in time.[30] IP addresses are assigned to all ***host*** computers that are connected to the internet as well as web servers. For instance, an IP address for a host on the network is of the format "203.0.113.10", wherein "203.0.113" is a network identifier for the network the host is connected to, while "10" is the identifier of the host itself.[31]

33. Consider the act of sending a physical letter from California to another individual who resides in New York. The IP address is equivalent to the street address of the individual down to the house number, street, city and state that directs the mailing service to the correct recipient address where the letter will be delivered.[32] Similarly, the ISP that handles the host's IP

[28] Computer Networking: A Top-Down Approach, p. 30.

[29] Computer Networking: A Top-Down Approach, p. 120.

[30] A domain name is usually represented by human understandable text, which in turn is translated into an IP address using a ***Domain Name System (DNS)***. See: Computer Networking: A Top-Down Approach, p. 160.

[31] Computer Networking: A Top-Down Approach, p. 160.

[32] One nuance to note here is that the IP address assigned by an ISP to a host can be determined dynamically (i.e., the IP address assigned to a host changes periodically based on availability and efficiency of the network) or can be static, meaning that the IP address for the host stays fixed and does not change over time.

address assignment is able to associate the IP address to a specific device connected to the internet.[33]

34. In practice, when a computer connects to the internet, the ISP assigns an IP address to the host. Then, as the user inputs a URL in a web browser, the domain name component is also translated into an IP address using a ***Domain Name System*** (DNS).[34] In this way, the web browser accurately identifies the address of the appropriate web server from which to request the required files. IP packets carrying a web request include the IP address of the user's device as the source address and the IP address of the web server as the destination address. The server then sends response packets containing the requested content (such as a webpage) back to the user by using the original source address as the destination. As a result, the IP address plays a key role in directing the required content to the appropriate computer, and back, without any confusion.

2) File Transfer, File Transfer Protocols, and Ports

35. The term ***downloading*** generally refers to a process by which a computer receives data from another computer, often a remote computer over a network. For example, when a computer requests and then receives a file from another computer on the internet, this is often referred to as downloading the file. ***Uploading*** refers to the opposite process, by which the computer transmits data to another computer such as a remote computer over a network. For example, if that same computer sends a file to another computer over the internet (such as in response to a request from the other computer), this is an example of uploading that file.

36. As discussed above, the TCP/IP protocol is the foundation on which data is transferred over the internet. However, specialized protocols are also developed on top of the TCP/IP protocol based on specific requirements of the data. For example, the HTTP protocol was initially designed specifically for transmitting hypertext files. Another common protocol, designed specifically for sharing files over the internet is the ***File Transfer Protocol*** (FTP).[35]

[33] In most cases, an external IP address is associated with a router that is used to connect the devices on a home or business network to the Internet.

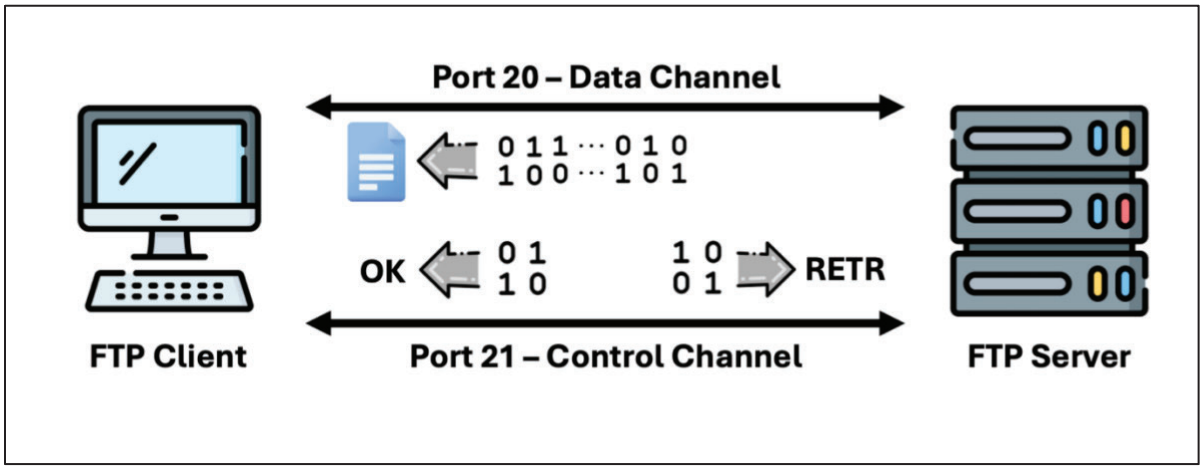
[34] Computer Networking: A Top-Down Approach, p. 161.

[35] Computer Networking: A Top-Down Approach, p. 116.

37. A majority of the interaction for data on the internet flows through a ***client-server model***, in which the interaction for accessing files is between (a) the client requesting data for download and (b) a centralized server that contains the requested data.[36] In this model, a user looking for particular information is the ***client*** and accesses the information from some centralized server or system, such as YouTube (for a cooking video), Amazon (for information about products), or Wikipedia (for an article from its large online encyclopedia).

38. Within the client-server model, protocols such as FTP enables a client requesting data to interact with the server containing the information by opening two channels of communication: control and data channels. Both these channels in turn connect with different ***ports*** on the server, which are the standardized endpoints for programs running on the server.[37] The ports enable a distinct flow of control information to be sent simultaneously but separately from a distinct flow of data between the same client and server. For instance, the control channel in FTP connects to port 21 on the FTP server, whereas the data channel connects to port 20, as shown below.[38]

Figure 1 - File Transfer Protocol Retrieve Process



39. The control channel communicates requests to the server such as "RETR" for retrieving a file from the server, or "STOR" for uploading a file to the server. In turn, the server responds with

[36] Computer Networking: A Top-Down Approach, p. 115-116.

[37] Computer Networking: A Top-Down Approach, p. 120-121.

[38] "WinZip | Download Your Free Trial," accessed February 5, 2025, https://www.winzip.com/.

the status of the file (e.g., if the file exists on the server) as well as requests for any additional information required from the client (e.g., password for accessing the file), on the control channel. The control channel is thus responsible for establishing, maintaining, and terminating the session between the client and the server.[39] On the other hand, the data channel is responsible for the actual transfer of file contents that have been broken down into small units and sent in data packets.[40] The compilation of all packets when received by the client results in the downloading of the file from the server on the internet.

40. Modern web browsers rely on HTTP or the more secure HTTPS protocols for downloading files.[41] Similar to FTP, HTTP and HTTPS clients send a "GET" command to the web server in order to download a file.[42] The server then responds with the data packets to port 80 for HTTP and port 443 for HTTPS,[43] which are then compiled and saved on the local machine by the browser. Even though the protocol for accessing the file on the web server differs, the actual data packets are still sent over TCP and IP, as discussed above.

41. All the protocols discussed thus far are based on a client-server model, in which the data to be acquired resides on a central server or system, giving rise to inherent limitations for communication of that data. The primary limitation is the dependency on a central server for delivery of required data files to multiple clients.[44,45] The development and maintenance of such a central server is often a costly undertaking and can lead to diminishing performance as a large number of clients request the same file from the central server, thereby competing for its available resources. When a significant number of clients request the same resource (e.g.,

[39] "What Is FTP? | Definition from TechTarget," Search Networking, accessed February 5, 2025, https://www.techtarget.com/searchnetworking/definition/File-Transfer-Protocol-FTP.

[40] "What Is FTP? | Definition from TechTarget," Search Networking, accessed February 5, 2025, https://www.techtarget.com/searchnetworking/definition/File-Transfer-Protocol-FTP.

[41] "What Is Downloading?," Search Networking, accessed February 5, 2025, https://www.techtarget.com/searchnetworking/definition/downloading.

[42] "IBM B2B Advanced Communications 1.0.0," March 5, 2021, https://www.ibm.com/docs/en/b2badv-communication/1.0.0?topic=api-download-file.

[43] "What Is a Computer Port? | Ports in Networking," accessed February 5, 2025, https://www.cloudflare.com/learning/network-layer/what-is-a-computer-port/.

[44] Computer Networking: A Top-Down Approach, p. 116.

[45] Computer Networking: A Top-Down Approach, p. 184.

during initial release of limited-edition products), the server can become overloaded, causing slowdowns and failures for some clients.[46] Another drawback of the client-server model is that a central server is responsible for retrieving and transmitting the data, and thus provides a "single point of failure," meaning that if the central server fails the data may become entirely inaccessible to all clients. And relatedly, any communication within the client-server model is restricted by the bandwidth available to the central server, which can result in the transfer of very large files taking a large amount of time, as the packets must be individually transmitted by the central server within the limited bandwidth available to it.

42. A ***peer-to-peer*** network architecture addresses these data transmission issues and is discussed in more detail in the following section.

3) Peer-to-Peer (P2P) Networks

43. A peer-to-peer (P2P) network operates in a similar fashion to the client-server model with one key difference: instead of a centralized server, in a peer-to-peer network generally speaking, every computer in the network can act as both a client and a server.[47] From a networking perspective, all computers within a P2P network are ***peers*** that can request specific information from other peers who possess it, or can provide the required information to other peers that are specifically requesting it.[48] This differs from the client-server model where only the centralized server holds the information that can be transmitted to connected clients. Thus, P2P networks are inherently decentralized networks.

44. Each computer that participates in a P2P network is referred to as a ***node***.[49] Whenever a node participates in a P2P network, it contributes to the overall bandwidth, content, storage, and processing power of the entire network. Typically, each client willing to participate in a P2P network runs dedicated software that, in accordance with the P2P protocol in question, initiates a process that allows the client to participate as a node on the P2P network. Essentially, the

[46] Computer Networking: A Top-Down Approach, p. 184.

[47] Computer Networking: A Top-Down Approach, p. 175-179.

[48] Computer Networking: A Top-Down Approach, p. 175.

[49] A peer is always a node, however a node can be a peer or a server client. See: Computer Networking: A Top-Down Approach, p. 62.

software initiates a virtual layer that runs on top of the existing network and is facilitated by the peers that are already participating in the network.[50] It is important to note however that the virtual network operates in much the same way as a physical network, with the key difference being that resources available on the network are contributed by the available nodes already participating in the peered virtual network. For instance, the information transfer between different nodes is still facilitated using a protocol such as TCP or UDP,[51] much like the client-server model,[52] but with an overlay of a P2P-specific communication protocol that defines how data exchanged between the peers is structured, exchanged and verified.

45. The bandwidth in a client-server network is restricted by the bandwidth of the central server. In contrast, the nodes in a P2P network act as both clients and servers, so each additional node contributes to the total bandwidth of the network. Similarly, while the web server in a client-server model possesses restricted processing power and is limited by the available computing resources of the individual server computer, in a P2P network each node contributes a portion of its available processing power to the overall network.

46. P2P networks are not a novel form of computer networking. During the 1960s, ARPANET, the first network capable of transmitting messages between two computers and the precursor to the modern internet,[53] was based on a P2P implementation.[54,55] P2P networks were used as the underlying network architecture of Skype (which stands for "Sky Peer to Peer") for voice

[50] "Azure Virtual Network Peering," November 19, 2024, https://learn.microsoft.com/en-us/azure/virtual-network/virtual-network-peering-overview.

[51] In some cases, the transfer of information may be facilitated by another protocol, "***User Datagram Protocol***" (UDP), another important component of the TCP/IP protocol suite. This protocol functions similarly to TCP, except that UDP is connectionless. Unlike TCP, clients do not need to establish and verify a connection before sending packets when using UDP. See: Computer Networking: A Top-Down Approach, p. 79.

[52] Jeffrey L Eppinger, "TCP Connections for P2P Apps." http://reports-archive.adm.cs.cmu.edu/anon/isri2005/CMU-ISRI-05-104.pdf.

[53] "ARPANET, Internet," LivingInternet (blog), accessed February 5, 2025, https://www.livinginternet.com/i/ii_arpanet.htm.

[54] A Network of Peers: Peer-to-Peer Models Through the History of the Internet - Peer-to-Peer [Book]," accessed February 5, 2025, https://www.oreilly.com/library/view/peer-to-peer/059600110X/ch01.html.

[55] "Steve Crocker Embodies Peer to Peer Architecture (P2P) as One of the Key Concepts of the ARPANET : History of Information," accessed February 5, 2025, https://historyofinformation.com/detail.php?id=878.

calls over the internet.[56,57] Concurrently, P2P networks and the protocols developed to support them (such as BitTorrent) are utilized for varied purposes such as distributing extremely large files over the internet, online gaming,[58] and supporting cryptocurrencies,[59] among other uses. My discussion of BitTorrent below provides more examples of the use of P2P networking in common real-world applications.

47. There are multiple different protocols that can be used to support P2P networking. The BitTorrent protocol is the P2P protocol that is at issue in this matter. Similar to the packet switching technique discussed above, when using the BitTorrent protocol, any large file is first converted to smaller pieces.

48. In practice, when a node using "BitTorrent" requests specific information (e.g. a request for a particular file), multiple peer nodes that have that information and are participating in the BitTorrent network may respond by providing a portion of the requested file. Since the pieces of the file may be collected from multiple nodes participating in the BitTorrent network (as opposed to a central server responding with all the relevant packets in the client-server model), the BitTorrent protocol can offer advantages with respect to both the availability of a particular file and also the overall speed and resiliency of the network. This is because unlike a client-server model, file distribution is not reliant on a single point of failure or any single bandwidth bottleneck. The entire network benefits by the number of peers that are connected on the network, as well as the number of peers that contain any specific information requested by a user.[60]

[56] "A Brief History of Skype - the Peer to Peer Messaging Service," accessed February 5, 2025, https://content.dsp.co.uk/history-of-skype.

[57] Salman A. Baset and Henning Schulzrinne, "An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol" (arXiv, December 5, 2004), https://doi.org/10.48550/arXiv.cs/0412017.

[58] "Peer-to-Peer Gaming," Medium, September 22, 2023, https://medium.com/tashi-gg/peer-to-peer-gaming-9991600c6707.

[59] "P2P Network — Bitcoin," accessed February 5, 2025, https://developer.bitcoin.org/devguide/p2p_network.html.

[60] Computer Networking: A Top-Down Approach, p. 178.

B. BitTorrent

49. Like HTTP discussed in **Section IV.A.2** above, BitTorrent is a communication protocol. BitTorrent protocol is used for P2P networks and allows nodes to communicate with one another as peers. Like the FTP communication protocol discussed above, the BitTorrent protocol is specialized to enable transfer of large files over the internet, specifically within P2P networks. As discussed below, based on unique characteristics and techniques used within the BitTorrent protocol, it can be efficient for the transfer of very large files over the internet, and thus, is used regularly in applications requiring downloading of large files.[61]

50. As discussed, an advantage of exchanging files over P2P networks is the increased efficiency afforded by the breakup of large files into small ***pieces*** that are distributed across many nodes participating in the P2P network. Some BitTorrent terminology has multiple context-dependent meanings. For simplicity, this section of my report describes the terms that I will use in this report to refer to different mechanisms within the protocol. Where possible, these are the same terms, with the same meanings, as are used in the Krein report.

51. As discussed in the Krein Report, "[o]nce a peer completed a download of the complete file, it could in turn function as a seed."[62] In line with this definition, I use the term ***seeder*** to describe peers that have completed the download of a file i.e., have a full copy of the original file. In contrast, the term ***leechers*** refers to peers that have begun the download and have no pieces, or that have downloaded only a portion of the complete file.[63] The term ***swarm*** is typically used to describe the collection of seeders and leechers for a particular file or set of files. Once a leecher receives all of the pieces of the complete file, it possesses a complete reassembled instance of the original file(s) and can then be designated a seeder. It is important to note that

[61] "About BitTorrent | Creator of the World's Leading P2P Protocol," BitTorrent, accessed February 5, 2025, https://www.bittorrent.com/company/about-us/ ("Before BitTorrent, file downloads were initiated from a centralized server or a single user (a peer), resulting in slow download speeds. The BitTorrent protocol addressed this limitation by enabling the download and upload of files between many users. Millions of users began to use the BitTorrent protocol to download and share files, and companies began to use the protocol to distribute data more efficiently. Today, the BitTorrent protocol powers a significant percentage of the world's Internet traffic each day. It isn't just the largest Peer-to-Peer network, it's the foundation of Web3, and one of the Internet's largest global communities. Proof that the technology is more relevant than ever, robust, and now driven by the power of blockchain.").

[62] Krein Report ¶119; See also: BitTorrent Limited, "Help Center - What Is Seeding?," BitTorrent, accessed February 8, 2025, https://www.bittorrent.com/en/support/solutions/articles/29000041669-what-is-seeding-.

[63] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf.

seeding is an optional process that can commence only after the download of the file has completed. The seeder can choose to stop seeding the data, for example, by not participating in the network once the download is finished.[64]

52. Central to the BitTorrent protocol is a ***torrent file*** (identified by a .torrent extension) which contains key information related to the file that is available on the P2P network. The torrent file does not contain the underlying content to be downloaded; it instead contains the metadata required to reconstruct and validate the payload, including an ordered list of names, and sizes of the files. To avoid ambiguity, I will refer to the underlying content of the target files to be downloaded as the torrent file's ***payload***.[65]

53. Each piece of a complete file that is exchanged over the BitTorrent protocol comprises a portion of some particular torrent file's payload. In addition to identifying the names and sizes of files, the torrent file also contains the ***hash values*** of each piece of the payload that will be downloaded by the client. A hash value is typically computed by a hashing function, which takes the input data and calculates a fixed-length string of characters known as a hash value. Hash values, thus, often serve two purposes in computing. First, hash values allow computers to verify the integrity of data they receive. Second, hash values are often used to uniquely identify data because the underlying hashing function is designed such that each different piece of data should have a different hash value.[66]

54. In BitTorrent, a ***piece hash*** value is derived for the contents of each individual piece of the torrent payload and stored in the torrent file.[67] When a new piece is received, a hash derived from its contents can be compared to the stored piece hash to detect tampering or corruption.

[64] "After a torrent job finishes downloading, you are highly encouraged to leave the torrent job seeding. Although the length of time that you should leave the file seeding is not defined, it is recommended that you share until the amount of data you upload reaches at least the same as the amount of data that you have download, also known as reaching a 1.0 ratio." See: BitTorrent Limited, "Help Center - What Is Seeding?," BitTorrent, accessed February 8, 2025, https://www.bittorrent.com/en/support/solutions/articles/29000041669-what-is-seeding-.

[65] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf.

[66] It is theoretically possible for hashing functions to generate collisions (*i.e.*, two different pieces of data generate the same hash value), but the possibility of this occurring is infinitesimally small.

[67] This assumes that pieces do not contain exactly the same replicated data.

If there are even slight deviations in the content of the piece that is downloaded, its hash will be different than the expected piece hash.

55. In addition to this piece hash, the torrent file also contains an ***info hash***, which is a hash calculated on all of the file and piece information contained within the torrent file.[68] The info hash is used within the BitTorrent network to identify a specific associated payload and also used by peers in their communications with the tracker, DHT network, and other peers in the swarm.

56. For the BitTorrent protocol, the hash values are calculated using the ***Secure Hash Algorithm - 1*** (SHA-1), a well-known hashing function, and the corresponding hash values are referred to as SHA-1 hashes.[69] As detailed in **Section IV.C.3** below and consistent with the use of hashing functions generally, the hash values are used to verify data integrity for each piece that is exchanged over the BitTorrent protocol, increasing the reliability of file distribution that uses this protocol.

57. Lastly, the torrent file contains the IP address of a ***tracker***.[70,71] A tracker is a server that contains information on the peers that are connected to a P2P network,[72] as well as the status of each peer, i.e., whether the peer has completed the download (i.e., is a seeder), or is still downloading (i.e., is a leecher). The main role of the tracker is for leechers engaged in a download of pieces to find peers that may have the piece that they require. A particular leecher can communicate with the tracker using the HTTP protocol by sending information about what torrent payload it wants to download, causing the tracker to respond with a list of contact information for peers that have that payload.[73] Alternatively, instead of a tracker, peers can

[68] "Bep_0003.Rst_post," accessed February 5, 2025, https://www.bittorrent.org/beps/bep_0003.html.

[69] "What Is SHA-1 and How Is It Used for Data Verification?" Lifewire, accessed February 5, 2025, https://www.lifewire.com/what-is-sha-1-2626011.

[70] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf.

[71] Torrent files may contain more than one tracker.

[72] Adele Lu Jia and Dah Ming Chiu, "Designs and Evaluation of a Tracker in P2P Networks," in 2008 Eighth International Conference on Peer-to-Peer Computing (2008 Eighth International Conference on Peer-to-Peer Computing (P2P), Aachen, Germany: IEEE, 2008), 227–30, https://doi.org/10.1109/P2P.2008.11.

[73] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf.

also utilize a ***distributed hash table*** (DHT), which functions as a decentralized way to access information about other peers available on the network.[74]

58. The final piece to understanding the success of the BitTorrent protocol is the choking and unchoking process that enables a leecher to manage its bandwidth and maximize its own download rate. The choke/unchoke algorithm helps a peer maximize its own download rate via the process that selects which peers to "unchoke." On a technical level, this decision is made by each peer by utilizing ***choking algorithms*** that periodically evaluate the peers to connect to. Leechers prioritize unchoking (i.e., allowing the exchange of pieces) from those peers where they can maximize their own download rate.[75]

59. Having described the relevant background for foundational concepts within the BitTorrent protocol, I detail in the next section the process of exchanging files using the BitTorrent protocol, highlighting how these mechanisms together provide an efficient and reliable method for the download and upload of large files over P2P networks.

C. Torrent Process

60. The torrent process is initiated by the client accessing the P2P network using specialized software. The users must first download and install a BitTorrent client on their computer in order to access the BitTorrent P2P networks.[76] While the BitTorrent protocol has a popular client with the same name,[77] i.e., "BitTorrent client," multiple other programs have also been developed that utilize the BitTorrent protocol to interact with peers in the torrent network. Some popular options include, uTorrent,[78] qBittorrent,[79] libtorrent,[80] among others. Since all

[74] Computer Networking: A Top-Down Approach, p. 181.

[75] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf.

[76] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf.

[77] BitTorrent Limited, "BitTorrent Classic | The Original Torrent Client for Desktop," BitTorrent, accessed February 5, 2025, https://www.bittorrent.com/products/win/bittorrent-classic-free/.

[78] BitTorrent Limited, "µTorrent (uTorrent) Classic | The Original Torrent Client," uTorrent, accessed February 5, 2025, https://www.utorrent.com/desktop/.

[79] "qBittorrent Official Website," accessed February 5, 2025, https://www.qbittorrent.org/.

[80] "Libtorrent," libtorrent, accessed February 5, 2025, https://libtorrent.org/index.html.

of these programs communicate using the same BitTorrent protocol, they are able to participate together in the same BitTorrent swarms.

61. The next step is for a user to locate a torrent file for the content they wish to download using BitTorrent. Usually, the torrent file can be located using websites that maintain a list of torrent files or by searching on a search engine web site to locate the desired torrent file.[81] Once the user downloads the torrent file, the torrent file can be opened using the torrent client on the user's computer. Opening a torrent file using a BitTorrent client software results in the creation of a BitTorrent session that allows the user's computer to communicate with other BitTorrent peers who are downloading/uploading the same payload (i.e. the specific payload described by the torrent file).

1) Connection to Peers

62. Within a torrent session, the first action taken by the torrent client is to locate other peers within the network for that particular torrent file. This is where the tracker is utilized to acquire information about other peers in the network who are participating in the same torrent payload. The leecher sends a request to the tracker passing the info hash of the torrent file as well as the port number on which the leecher is expecting to receive the downloads.[82] The tracker responds with a list of randomly selected peers (including their IP addresses and a peer id) within the network who are currently active for the desired info hash payload.[83] Alternatively, leechers can also access information about other peers by utilizing DHTs. When using DHTs, each node in a swarm contains information about a subset of the other nodes in the P2P network, and messages can be passed from node to node to traverse the network and obtain information or files from other nodes in the network. Similar to utilizing a tracker, when a leecher requests to download a specific payload, a request with the info hash for that payload is sent to the DHT network, which then responds with information about peers who are participating in the BitTorrent network for that payload.[84] In this way, by utilizing either a

[81] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf.

[82] "Bep_0003.Rst_post," accessed February 5, 2025, https://www.bittorrent.org/beps/bep_0003.html.

[83] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf.

[84] Ryan McCarthy, "What Is DHT and How Does It Work for BitTorrent?" December 15, 2022, https://www.downloadprivacy.com/how-to-torrent/dht.

tracker or a decentralized DHT system, a leecher is able to identify peers that have the desired payload.

63. Once the leecher receives a list of peers that are participating in a swarm for the desired payload, the next step is to connect with the peers. Each connection is initiated in the form of a protocol-defined handshake between two nodes. Key steps in the handshake include (i) an exchange that confirms both peers are running client software that uses the BitTorrent protocol and participating in a swarm for a specific payload, which is done by transmitting the info hash associated with a particular payload as a part of the initial protocol handshake, and (ii) having the expected peer id, which is checked by validating a peer's reported id against the peer id the tracker provided.[85] If both sides do not send the same info hash value, or a peer id does not match, the peers sever their connection.[86] Next, the peers begin exchanging messages. As a part of this exchange, the peers first identify what pieces of the payload they already have, and later what pieces they are seeking.

2) Downloading Pieces

64. Once connected to other peers via the handshake process described above, a leecher can request different pieces of the payload from multiple peers simultaneously in order to boost the chances of finding peers that can provide the pieces it wants.[87] It does so by sending out piece request messages to peers that have indicated they have pieces that the leecher needs. As the peers respond, the requesting peer can download different pieces from different peers, at the same time.

65. There are further optimizations that are conducted by the BitTorrent protocol at this stage, such as (i) breaking up of pieces into sub-pieces,[88] (ii) sending multiple queued requests for sub-pieces to the peers,[89] as well as (iii) organizing the pieces that are prioritized for download by

[85] "Bep_0003.Rst_post," accessed February 5, 2025, https://www.bittorrent.org/beps/bep_0003.html.

[86] "Bep_0003.Rst_post," accessed February 5, 2025, https://www.bittorrent.org/beps/bep_0003.html.

[87] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf.

[88] Typically of 16KB each. In this report, when I refer to a piece, I only refer to the full piece, instead of the sub-pieces that are created by the BitTorrent client.

[89] Otherwise referred to as TCP pipelining, a process to ensure that the available bandwidth is fully utilized for downloading accessing different sub-pieces.

the leecher.[90] The first and the second optimizations are implemented such that multiple portions of a piece can be received from different peers in parallel, thereby reducing the time to download a given piece. Lastly, the selection and prioritization of the pieces to download is based on the status of the downloaded file. When a download is initiated, a random piece of the file is typically prioritized for download. After the first full piece is downloaded, the BitTorrent protocol may switch to a "rarest-first" piece selection policy. As its name implies, a "rarest-first" policy prioritizes selection for download of the rarest pieces, (*i.e.*, pieces that are available from the fewest number of connected peers). Prioritizing rare pieces over pieces that are more commonly available helps ensure that additional copies of the rare piece become available within the swarm more quickly.[91]

3) Completion of Download

66. Finally, when a piece is received by a peer such that all the sub-pieces for that piece have also been downloaded, a hash check is conducted by comparing the hash calculated from the received piece with the corresponding piece hash provided within the torrent file (which was downloaded at the beginning of the process).[92] If the hashes match, it is verified that the downloaded piece is received as expected, i.e., with the same contents and devoid of any corruption or outside tampering. Once all the sub-pieces and pieces are received, the BitTorrent client effectively consolidates all the constituent pieces into the completed payload.[93] Once the entire payload has been successfully downloaded, the BitTorrent client also reports to the tracker that it has completed the download so that the tracker knows that it is now a seeder.

67. Once the download is fully completed, by default the leecher becomes a seeder.[94] At this point, the seeder can choose to remain a part of the swarm and seed data to other peers. However, this is not a requirement in the protocol and BitTorrent clients, as well as the user, can terminate

[90] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf.

[91] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf.

[92] "Bep_0003.Rst_post," accessed February 5, 2025, https://www.bittorrent.org/beps/bep_0003.html.

[93] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf.

[94] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf.

seeding once the file is downloaded, terminating the node's connection with the torrent network and thereby preventing the uploading of any pieces to other peers.[95]

68. In the next section, I highlight contemporary uses of the protocol for the exchange of information, by entities such as universities, open-source program distributions, and government organizations.

D. Use Cases of BitTorrent Protocol

69. Since its introduction, BitTorrent has garnered immense support and has become the most popular protocol for peer-to-peer networking.[96,97] Garnering support within the community for exchanging open-source software, i.e., software that is distributed with permissive licenses to be used and developed upon by other users,[98] BitTorrent has been utilized extensively for distribution of the popular Linux operating system.[99] Owing to its open source nature Linux has overtaken the popular Windows operating system across variety of devices such as servers,[100] smartphones,[101] and in-vehicle entertainment systems,[102] all while being distributed by utilizing the BitTorrent protocol.[103]

[95] BitTorrent Limited, "Help Center - What Is Seeding?," BitTorrent, accessed February 8, 2025, https://www.bittorrent.com/en/support/solutions/articles/29000041669-what-is-seeding-.

[96] Bram Cohen, "Incentives Build Robustness in BitTorrent." https://www.bittorrent.org/bittorrentecon.pdf.

[97] Jahn Arne Johnsen and Lars Erik Karlsen, "Peer-to-Peer Networking with BitTorrent," https://web.cs.ucla.edu/classes/cs217/05BitTorrent.pdf.

[98] "What Is Open Source Software? | IBM," July 29, 2021, https://www.ibm.com/think/topics/open-source.

[99] "If any group has embraced the possibilities and power of BitTorrent for distribution, it is the Linux community. Virtually every distribution is available via torrent download." See: BitTorrent Limited, "Introducing µTorrent Server for Linux," BitTorrent, accessed February 5, 2025, https://www.bittorrent.com/blog/2010/09/02/introducing-µtorrent-server-for-linux.

[100] "Linux or Windows Servers? Which One's Better | Volico," Miami and Broward Colocation | Volico Data Centers (blog), May 20, 2024, https://www.volico.com/linux-or-windows-servers-whats-the-difference-and-whichones-better/.

[101] Android OS runs on a Linux Kernel. See: "Linux or Windows Servers? Which One's Better | Volico," Miami and Broward Colocation | Volico Data Centers (blog), May 20, 2024, https://www.volico.com/linux-or-windows-servers-whats-the-difference-and-whichones-better/.

[102] Robert Huntley, "Linux Gaining Ground in Automotive," EE Times Europe, September 3, 2024, https://www.eetimes.eu/linux-gaining-ground-in-automotive/.

[103] BitTorrent Limited, "Introducing µTorrent Server for Linux," BitTorrent, accessed February 5, 2025, https://www.bittorrent.com/blog/2010/09/02/introducing-µtorrent-server-for-linux.

70. Further, as discussed in **Section IV.B** and **IV.C**, BitTorrent provides an effective and efficient protocol for the transfer of very large files over the internet. BitTorrent is popular among varied institutions such as universities, and governmental and non-governmental organizations. For instance, Florida State University utilized the BitTorrent protocol for distribution of scientific datasets, citing the ability for users to "easily share large data sets using BitTorrent."[104] Additionally, NASA utilized BitTorrent for the distribution of data related to the "visible earth" project for distributing images of Earth for researchers and enthusiasts alike.[105]

71. Owing to its ability to distribute large files quickly, BitTorrent has also been utilized for distribution of security patches, as well as updates to enhance gaming experience for users. For instance, Blizzard Entertainment developers of the popular game "World of Warcraft," utilized BitTorrent for the distribution of patches to the game.[106] World of Warcraft is estimated to have over 160 million players.[107]

72. Since its inception in the early 2000s, BitTorrent has been influential and has been utilized by millions of users for varied purposes including facilitating research, distribution of open-source software, and even promotion of security through distribution of software patches. Examining these diverse use cases provides important context for understanding how BitTorrent operates in practice.

73. Building on this foundation of the BitTorrent protocol and its use cases, the following sections will explain various errors and misstatements in the Krein Report regarding the seeding of data within BitTorrent and the way in which Meta used it to download certain datasets.

[104]"HPC Data Repository," April 2, 2013,https://web.archive.org/web/20130402200554/https://www.hpc.fsu.edu/index.php?option=com_wrapper&view=wrapper&Itemid=80.

[105] "Nasa Is Using BitTorrent for Their 'Visible Earth'Project * TorrentFreak," accessed February 5, 2025, https://torrentfreak.com/nasa-is-using-BitTorrent-for-their-visible-earthproject/.

[106] Jahn Arne Johnsen and Lars Erik Karlsen, "Peer-to-Peer Networking with BitTorrent," https://web.cs.ucla.edu/classes/cs217/05BitTorrent.pdf.

[107] "Server Population & Player Count," MMO Populations, accessed February 5, 2025, https://mmo-population.com/r/wow.

V. OVERVIEW OF THE RELEVANT SOURCE CODE AND AT-ISSUE DATASETS

74. I have analyzed the two instances in which it has been alleged in the Krein Report that BitTorrent was used by Meta to download datasets claimed to be at issue in this case: (i) in 2023 to download one portion of a dataset known as Library Genesis (LibGen), and (ii) in about April 2024 to download certain datasets from Anna's Archive. In this section, I first respond to the statements in the Krein Report that contain misstatements about the use of source code scripts to perform torrent downloads. I then present the list of datasets actually downloaded by Meta through torrents, demonstrating how the Krein Report inflates the count of the datasets downloaded by Meta using the BitTorrent protocol.

A. The Krein Report Misrepresents Meta's Torrent Source Code

75. The Krein Report discusses the "**download_trnts.py**" code, which based on the evidence I have reviewed, is a Python script that was used by Meta engineer Nikolay Bashlykov in 2023 to download a portion LibGen, and in particular, a portion of that library known as "scimag" that contains scientific articles.[108] As I will explain below in **Section V.B.2**, Plaintiffs' books are not contained in this "scimag" library. There is no evidence that this script was used by Mr. Bashlykov to download any other portions of LibGen or any other dataset, including the fiction subset that Plaintiffs have alleged contains some of their copyrighted works.[109]

76. The evidence indicates, to the contrary, that Mr. Bashlykov used a direct download technique that did not involve the use of the "**download_trnts.py**" script or otherwise use BitTorrent, to download the Fiction and Scitech portions of LibGen. The Krein Report appears to acknowledge as much, stating that the "Meta source code file **download_libgen_direct.py** defines functionality for directly downloading LibGen."[110] It then states: "The file includes a source code comment stating, '`Script to load the libgen library (scitech/fiction) using a direct download link (`***<u>`not torrenting`</u>***`).`'"[111] I have reviewed this source

[108] "Index of /Scimag/Repository_torrent," accessed February 8, 2025, https://libgen.is/scimag/repository_torrent/.

[109] Opening Expert Report of Cristina Videira Lopes, Ph.D., Jan. 10, 2025, ¶¶102, 163 (alleging that LibGen contains 44 at-issue works), 164 (alleging that another LibGen subset contains 34 at-issue works).

[110] Krein Report ¶108.

[111] Krein Report ¶108 (emphasis added).

code script and agree that it does not use the BitTorrent protocol.[112] This is corroborated by other contemporaneous internal Meta documents stating that Mr. Bashlykov used a direct download technique for downloading the scitech and fiction portions of LibGen,[113,114] and which was again confirmed to me during my interview with Mr. Bashlykov.[115] Because these portions of LibGen were not downloaded using torrents, their download could not have created any possibility of upload of that data to other peers via BitTorrent.

77. The Krein Report also acknowledges that the "**download_trnts.py**" script "includes a comment explaining that the script '`downloads` ***`scimag`*** `using torrent files.`'"[116] The Krein Report does not cite any evidence that the "**download_trnts.py**" script was ever used to download any other portion of LibGen aside from Scimag. It at best speculates that the "**download_trnts.py**" script *could* have been used to download other portions of LibGen but cites no evidence that this ever occurred. Krein acknowledges that the **download_trnts.py** script would have to be modified before it could be used to accomplish his hypothetical downloads. For example, the Krein Report claims that "the **download_trnts.py** script is designed with swappable variables," such that switching the directory from "scimag to, say, scitech or fiction" would have enabled a torrent download of the scitech or fiction libraries.[117] The Krein Report also states that "the source code has been architected for use with not just 'scimag', but also explicitly for use with scitech and fiction as well."[118] But the Krein Report does not present any evidence that any of these hypothetical modifications to the "**download_trnts.py**" script ever occurred. The Krein Report further speculates that the TORRENT_FOLDER variable in the "**download_trnts.py**" script could have been changed

[112] META-KADREY-SC-000197.

[113] Meta_Kadrey_00168648 at Meta_Kadrey_00168656 (entry for 06/16/2023: "LibGen next steps […] probably would need to disclose that we torrented scimag – it wasn't an option to direct-link download it as I did for scitech/fiction, [because] it's very large….").

[114] Meta_Kadrey_00211852 (01/05/2024, 03:15:52 message: "nikolay clarified that it was mostly direct download, only the scimag data was torrented").

[115] Interview with Meta engineer, Nikolay Bashlykov.

[116] Krein Report ¶120 (emphasis added).

[117] Krein Report ¶121.

[118] Krein Report ¶122.

to point to a directory called "`scitech_trnt_files`," or "`fiction_trnt_files`,"[119] but cites no evidence that this ever occurred or that these directories ever existed. Further. Dr. Krein states with no evidence that the DOWNLOAD_DIR variable in the August 2024 "**download_trnts.py**" script "has been set to various paths in the past" even when the March 2024 version he printed also contains no evidence of such past values.

78. I understand that the "**download_spark.py**" script was used in April 2024 to download certain datasets from Anna's Archive.[120] The Krein Report also misrepresents this script, stating "[i]n addition to Meta's documents, its source code also shows LibGen ***fiction*** being torrented from Anna's Archive."[121] The Krein Report claims that because "Meta's script for torrenting from Anna's Archive includes an 'Example command,' which specifies a '`dataset_name`' of '`libgen_rs_fic`,' as well as an '`input_path`' […] and an '`output_dir`'", the mentioned directories are where "LibGen fiction (i.e., '`libgen_rs_fic`') is the data torrented."[122] But as I detail further in **Section V.B** below, the actual files downloaded using this script do not contain LibGen Fiction.[123] I also note here that the "Example command" that Krein cites for this file is present only in the form of comments, not as executable code. In evaluating the text of comments, it is important to understand that they are discretionary annotations that a programmer can record in a script or program. They are not functional code, and they do not control any aspect of the script's actual processing.

79. The Krein Report relies on static and isolated source code, variables, and comments within source code to suggest what datasets were downloaded via torrenting at Meta, often ignoring contemporaneous internal documents. Below, in **Section V.B** I outline the datasets that were downloaded at Meta.

[119] Krein Report ¶121.

[120] Interview with Meta engineer, Xiaolan Wang.

[121] Krein Report ¶126.

[122] Krein Report ¶126.

[123] Interview with Meta engineer, Xiaolan Wang.

B. The Krein Report Misrepresents the Datasets Downloaded by Meta

80. As I outlined in **Section V.A** above, the Krein Report makes vague assertions regarding the download of datasets at Meta, claiming "Meta's Source Code Includes Scripts to Torrent LibGen,"[124] and "Meta's Source Code Includes Scripts to Torrent Anna's Archive."[125] Below, I discuss which datasets were downloaded at Meta, and narrow the scope of Meta's torrenting to April 2024 because, as noted, the "scimag" portion of LibGen (the only torrents downloaded in 2023) does not contain Plaintiffs' works.

1) Meta Did Not Torrent All of LibGen

81. The Krein Report's claim that "Meta used BitTorrent to download at least one copy of LibGen"[126] is incorrect. As demonstrated in **Section V.A** above, and confirmed by contemporaneous documents,[127,128] as well as the Meta engineer responsible for the torrent download,[129] only the "scimag" library portion of LibGen was downloaded in 2023 at Meta using the BitTorrent protocol. I have analyzed this subset of LibGen in **Section V.B.2** and confirmed that it does not contain Plaintiffs' works. Therefore, the "**download_trnts.py**" script that the Krein Report includes in its discussion of the download of LibGen, and Meta's 2023 torrenting efforts more generally, are not relevant to the at-issue works in this case.

[124] Krein Report § 9.3.2.

[125] Krein Report § 10.2.2.

[126] Krein Report ¶117.

[127] Meta_Kadrey_00168648 at Meta_Kadrey_00168656 (entry for 06/16/2023: "LibGen next steps […] probably would need to disclose that we torrented scimag – it wasn't an option to direct-link download it as I did for scitech/fiction, [because] it's very large….").

[128] Meta_Kadrey_00211852 (01/05/2024, 03:15:52 message: "nikolay clarified that it was mostly direct download, only the scimag data was torrented").

[129] Interview with Meta engineer, Nikolay Bashlykov.

82. To determine the relevant datasets torrented by Meta in 2024 that could have contained one or more of Plaintiffs' works, I reviewed internal Meta documents and interviewed the Meta engineer responsible for the effort.[130][131] The relevant datasets are listed in **Table 1** below.

<u>Table 1 – Datasets Download by Meta Using BitTorrent</u>

Dataset
Portions of Libgen.rs Non-Fiction "scitech"[132,133]
Internet Archive (IA) [134,135]
Z-Library (ZLib) [136,137]

83. These datasets were indexed by Anna's Archive, a searchable online database that aggregates links to books, academic papers, and other digital content from various other libraries, including LibGen and ZLib. Launched in 2022, Anna's Archive describes its purpose as aiming to provide universal access to knowledge by indexing and preserving sources to access materials.[138] Unlike traditional online libraries, Anna's Archive does not host files itself but instead acts as a meta-search engine, directing users to other sources where they can download

[130] Interview with Meta engineer, Xiaolan Wang; *see also* Meta_Kadrey_00108336 (internal chat conversation listing Internet Archive, Z-Library, portions of LibGen, DuXiu as datasets being downloaded). DuXiu is a dataset containing Chinese language works, and I am not aware of any Chinese language books being among the Plaintiffs' copyrighted works that were allegedly infringed in this case.

[131] Xioalan Wang, a Meta engineer that I interviewed who conducted the 2024 torrent download at Meta created lists of downloaded files, which I reviewed. She created the list by executing a recursive directory command to create a list of files in the AWS storage associated with the download that occurred in 2024.

[132] Interview with Meta engineer, Xiaolan Wang.

[133] meta_nonfic_downloads.txt (This file contains a listing of all LibGen Non-Fiction works Meta downloaded on the AWS instance in 2024).

[134] Interview with Meta engineer, Xiaolan Wang.

[135] meta_ia_downloads.txt (This file contains a listing of all IA works Meta downloaded on the AWS instance in 2024).

[136] Interview with Meta engineer, Xiaolan Wang.

[137] meta_zlib_downloads.txt (This file contains a listing of all ZLib works Meta downloaded on the AWS instance in 2024).

[138] "Frequently Asked Questions (FAQ) - Anna's Archive," accessed February 5, 2025, https://annas-archive.org/faq.

requested materials.[139] Anna's Archive indexes instances of individual books across the datasets that can be sourced from the library. Using its search engine and searching for the book titles and authors provided by the Plaintiffs, I was able to identify all individual torrent files from the LibGen, ZLib, and IA datasets that contain the Plaintiffs' works.

84. LibGen is a digital library that contains millions of books, papers, and textbooks that users can find and download for free.[140] The content is divided by genre, such as fiction, or non-fiction, that is in-turn available to users for downloading content by genre. Following this division, the "Libgen.rs Fiction" dataset mostly contains fictional works such as fictional novels, children's books, etc., the "Libgen.rs Non-Fiction" dataset contains non-fictional works, such as biographies, textbooks, etc., and the "Libgen.rs scimag" dataset contains content from scientific journals, academic papers, etc., as noted. Similar to LibGen, Z-Library ("ZLib") is also a digital library containing millions of books and articles.[141] Finally, Internet Archive ("IA") is a compendium of content available through the internet, since its inception, such as web pages, images, audio recordings, etc.[142] For instance, old versions of a web page may be contained within Internet Archive and can be accessed using the Wayback Machine, which preserves and provides access to archived web pages.[143] The IA data downloaded in this matter also contains books.

2) The Scimag Portion of LibGen Does Not Contain Plaintiffs' Works

85. To verify the contents of the "scimag" portion of the LibGen dataset, I manually searched and retrieved search results from Anna's Archive across all works in all at-issue datasets. I searched for each of the Plaintiffs' works on Anna's Archive, using Anna's Archive search

[139] "Frequently Asked Questions (FAQ) - Anna's Archive," accessed February 5, 2025, https://annas-archive.org/faq.

[140] "Library Genesis," Library Genesis Guide, accessed February 5, 2025, https://librarygenesis.net/.

[141] "About Us | Z-Library. Download Books for Free. Find Books," accessed February 5, 2025, https://z-lib.io/pages/about-us.

[142] "About IA," accessed February 5, 2025, https://archive.org/about/.

[143] "What Is Wayback Machine? | Definition from TechTarget," WhatIs, accessed February 5, 2025, https://www.techtarget.com/whatis/definition/Wayback-Machine.

engine,[144] and extracted URLs and other metadata (such as the MD5 hash[145] of each work, the associated torrent filename, and the associated file size for the Plaintiff work occurrence) for all works that matched. Additionally, I searched for different variations of the Plaintiffs' named works, including those in foreign languages,[146] to ensure I captured the exhaustive list of all occurrences of these works indexed by Anna's Archive.

86. The results of this search confirm that no at-issue Plaintiffs' works exists within "scimag." The name of each torrent file within the "scimag" portion of LibGen is formatted to begin with the "sm_" identifier, indicating that the works that are the payload of that torrent file belong to the scientific articles collection of LibGen.[147] No at-issue Plaintiffs' works are associated with any torrent file that begins with the "sm_" identifier, demonstrating that "scimag" does not contain Plaintiffs' works. This is confirmed through the analysis of the torrent files associated with each occurrence of Plaintiffs' works in the search results, which is described in **Appendix B**.

87. The relevant at-issue datasets that contain Plaintiffs' works downloaded at Meta only include IA, ZLib, and a portion of LibGen Non-Fiction representing Scitech. The "scimag" portion of LibGen is not a relevant at-issue dataset, as it does not contain Plaintiffs' works. Accordingly, the following analysis focuses solely on the confirmed relevant at-issue datasets,

VI. IT IS HIGHLY UNLIKELY THAT META SEEDED PLAINTIFFS' WORKS

88. The Krein Report does not present any evidence that Meta ***actually*** seeded any data, let alone that Meta seeded data comprising Plaintiffs' works. The Krein Report discusses only the ***potential*** for seeding, but this does not demonstrate any ***actual*** seeding.[148] As I demonstrate

[144] "New Search - Anna's Archive," accessed February 8, 2025, https://annas-archive.org/search.

[145] An MD5 hash is a unique identifier for a book on Anna's Archive. The search results are extracted in the form of "https://annas-archive.org/md5/<MD5 Hash>." Different versions of a book may exist in the same or different datasets downloaded by Meta. For this analysis, I consider all MD5 hashes that are associated with all different versions of the Plaintiffs' works that are accessible through Anna's Archive.

[146] Search results indexed by Anna's Archive contained Plaintiffs' works in languages other than English, including: Chinese, Czech, Dutch, French, German, Italian, Korean, Polish, Portuguese, Russian, Spanish, and Turkish.

[147] "Index of /Scimag/Repository_torrent," accessed February 8, 2025, https://libgen.is/scimag/repository_torrent/.

[148] Krein Report §§ 9.3.4, 10.2.4.

in **Sections VI.C.2** below, Plaintiffs' works constitute a vanishingly small percentage of the downloaded datasets. In light of the evidence that (i) Meta implemented safeguards to inhibit seeding, (ii) only a small proportion of the datasets comprise the Plaintiffs' works, as well as (iii) the large number of events that must match up (as discussed in **Section VI.C.4** below) for the potential seeding of any of the Plaintiffs' works to have occurred, the Krein Report only presents conjecture, without any evidence, to claim that Meta seeded any of the Plaintiffs' works. In my opinion, it is highly unlikely that Meta seeded any of the Plaintiffs' works.

A. Seeding of a Torrent Can Only Occur After the Download of a Torrent is Complete

89. The Krein Report makes several general assertions regarding the BitTorrent protocol using imprecise language, specifically around seeding, and concludes that files downloaded by Meta "become a seed for distribution to other peers in the network" once a given file has been fully downloaded.[149] However, the Krein Report does not acknowledge the fact that seeding is an optional process for a peer that has become a seeder, and only initiated by the BitTorrent client after the download for the entire payload (not just a single file) is **finished**. In this section, I explain the difference between peers and seeders to demonstrate that, due to the safeguards in place to prevent seeding and the many factors that would have needed to occur for seeding, it is extremely unlikely that Meta acted as a seeder for any of the Plaintiffs' works.

90. As discussed in **Section IV.B** and further defined on the BitTorrent website: "Seeding means sharing a file(s) with other peers. After a torrent job finishes downloading, if you leave the torrent job seeding, it uploads the file(s) to other peers so they can enjoy them too."[150] I agree with this definition and note that the Krein Report provides the exact same definition.[151] Conversion of a peer into a seeder, by definition, can only occur after the peer has completed the download of the entire payload. Seeding is an optional process in the BitTorrent protocol, as the source that the Krein Report cites also acknowledges; although many sites related to BitTorrent encourage seeding and consider it to be a social norm or informal rule of etiquette,

[149] Krein Report ¶¶127, 168.

[150] BitTorrent Limited, "Help Center - What Is Seeding?," BitTorrent, accessed February 8, 2025, https://www.bittorrent.com/en/support/solutions/articles/29000023347-what-is-seeding-.

[151] Krein Report ¶119.

there are no technical requirements that a peer remain active on the BitTorrent network as a seeder after it has completely downloaded all of the pieces of the torrent file's payload.[152]

91. The Krein Report alleges that Meta's download of the torrent payloads means that it subsequently acted as a seeder for the downloaded files. However, as I describe in **Section VI.B.2** below, the code implemented by Meta terminates the torrenting session as soon as it detects that the download has completed (i.e., within no more than 60 seconds of completion of the full torrent file download.) This action would disconnect any open peer connections, thereby significantly limiting the likelihood that Meta acted as a seeder for any of the torrented files.

92. Further, the implemented network configurations (as discussed in **Section VI.C.1**) would have prevented any new leechers from initiating a connection with Meta's instance during this (or any other) time window. Thus, the Krein Report's assumption that the downloading of a file via BitTorrent would have automatically led to seeding of that file is unfounded. As I discuss in the following sections, the safeguards implemented by Meta would have further significantly reduced the chance that Meta seeded any content, including any pieces containing the Plaintiffs' works.

B. Meta's Implemented Safeguards Rendered the Possibility of Seeding Highly Unlikely

93. The Krein Report repeatedly misrepresents the functionality of the source code implemented by Meta, specifically asserting that "is_seed() function […] checks whether a given file has been fully downloaded, at which point the file has become a seed for distribution to other peers in the network."[153] In this section, I respond to the Krein Report by highlighting the actual implementation of the code utilized, showcasing that the "**is_seed()**" function is used for checking the status during download, and subsequent removal of the torrent once the download is finished. I first provide an overview of the general torrent download functionality within

[152] "After a torrent job finishes downloading, you are highly **encouraged** to leave the torrent job seeding. Although the length of time that you should leave the file seeding **is not defined**." See: BitTorrent Limited, "Help Center - What Is Seeding?," BitTorrent, accessed February 8, 2025, https://www.bittorrent.com/en/support/solutions/articles/29000041669-what-is-seeding-.

[153] Krein Report ¶¶127, 168.

Meta's download scripts "**download_trnts.py**" and "**download_spark.py**". Interaction with the libtorrent library, and the BitTorrent protocol, is largely similar across both scripts, with differences limited to the manner in which parallel processing is implemented. I then outline how the safeguards implemented by Meta substantially limited the possibility of seeding Plaintiffs' works.

1) Overview of Meta's Torrent Download Source Code

94. As mentioned above, during my review of Meta's source code, I identified two source code files that manage BitTorrent downloads: "**download_trnts.py**" [154] and "**download_spark.py**,"[155] which were used to perform the torrent downloads, respectively, in 2023 (for "scimag" portions of LibGen) and in 2024 (for portions of Anna's Archive). Although only the latter script was used to download content including Plaintiffs' works using the BitTorrent protocol, my discussion will cover both scripts as they use the same logic for downloading large datasets via the BitTorrent client, in this case, libtorrent, a well-known open-source programming library for implementing the BitTorrent protocol.[156] The "**download_trnts.py**" script utilizes the "**download_torrent**" function, whereas the "**download_spark.py**" script utilizes an identical implementation in the "**_download**" function, described in more detail below, to initiate and handle the torrenting process. The differences between the two scripts lie in the way this common torrent mechanism is multitasked, such that "**download_trnts.py**" is designed to be executed multiple times in parallel on a single computer, whereas "**download_spark.py**" leverages the Python library[157] PySpark[158] to distribute processing multiple times in parallel across multiple computers, reducing overall download time.

[154] META-KADREY-SC-000202.

[155] META-KADREY-SC-000212.

[156] "Libtorrent," libtorrent, accessed February 5, 2025, https://www.libtorrent.org/.

[157] An external Python library is a collection of pre-written code modules created by others that can be installed and used in Python projects to add specific functionality or features without writing them from scratch.

[158] "PySpark Overview — PySpark 3.5.4 Documentation," accessed February 5, 2025, https://spark.apache.org/docs/latest/api/python/index.html.

95. As highlighted above, the "**download_torrent**" and the "**_download**" functions are the only components in the two source code files that interface with libtorrent and ultimately are responsible for downloading the torrent.[159,160] The implementation for handling the torrent files and downloading the payload are substantially identical between the two, with only minor differences as described below. The following steps are performed for each torrent file in both "**download_torrent**" and "**_download**" functions, in "**download_trnts.py**" [161] and "**download_spark.py**"[162] respectively, for downloading a particular torrent:

   a. A new libtorrent session is initiated, which serves as the central controller for managing torrent-related activities, including downloading files.[163,164,165]

   b. The torrent file information, such as the torrent info hash and corresponding piece hashes, are loaded into the machine/instance using libtorrent's "**torrent_info**" function.[166,167,168]

   c. A torrent is added to the libtorrent session using libtorrent's "**add_torrent**" function, providing the torrent file information and the path to the directory where the data downloaded from the torrent should be saved.[169] As soon as the "**add_torrent**" function is executed, libtorrent uses the BitTorrent protocol to begin participating in the BitTorrent network to obtain the indicated torrent payload files. This activity continues until the "**remove_torrent**" function is executed or the

[159] META-KADREY-SC-000202. Function "download_torrent" on line 38.

[160] META-KADREY-SC-000212. Function "_download" on line 78.

[161] META-KADREY-SC-000202. Function "download_torrent" on line 38.

[162] META-KADREY-SC-000212. Function "_download" on line 78.

[163] META-KADREY-SC-000202. Line 39.

[164] META-KADREY-SC-000212. Line 74.

[165] "Libtorrent," libtorrent, accessed February 5, 2025, https://libtorrent.org/reference-Session.html#session.

[166] META-KADREY-SC-000202. Line 42.

[167] META-KADREY-SC-000212. Line 78.

[168] "Libtorrent," libtorrent, accessed February 5, 2025, https://libtorrent.org/reference-Torrent_Info.html#torrent_info.

[169] "Libtorrent," libtorrent, accessed February 5, 2025, https://libtorrent.org/reference-Session.html#add-torrent-async-add-torrent.

script stops running. [redacted].[170,171]

[redacted]

[redacted]

[redacted]

[170] META-KADREY-SC-000202. Line 43.

[171] META-KADREY-SC-000212. Line 83.

[172] META-KADREY-SC-000202. Line 55.

[173] META-KADREY-SC-000212. Line 92.

[174] META-KADREY-SC-000202. Line 47.

[175] META-KADREY-SC-000212. Line 85.

[176] "Libtorrent/Src/Torrent_handle.Cpp at 57fd4e452eff0466957460b1608b96719599076e · Arvidn/Libtorrent," GitHub, accessed February 6, 2025, https://github.com/arvidn/libtorrent/blob/57fd4e452eff0466957460b1608b96719599076e/src/torrent_handle.cpp.

[177] Polling occurs when the source code repeatedly queries or checks the status of something at regular intervals.

[178] META-KADREY-SC-000202. Lines 48-50.

[179] META-KADREY-SC-000212. Lines 86-90.

[180] "**remove_torrent()** will close all peer connections associated with the torrent and tell the tracker that we've stopped participating in the swarm. This operation cannot fail." See: "Libtorrent," libtorrent, accessed February 5, 2025, https://libtorrent.org/reference-Session.html#remove-torrent.

g. At this point, the torrent download process is complete and the function responsible for downloading the torrent ends. This means the libtorrent session object is automatically destroyed, and no seeding of the downloaded content to other peers can occur.

[redacted]

97. Every step described above is implemented within both "**download_trnts.py**" and "**download_spark.py**." [redacted]

2) Meta's Source Code Removes Torrents from the Session Within No More Than 60 Seconds of Download Completion

98. A key component of the torrent download process in Meta's environment is the "**is_seed**" loop that monitors the status of the download, removing the torrent from the active session within no more than 60 seconds of completion. The body of the Krein Report asserts that the

[181] "**remove_torrent()** will close all peer connections associated with the torrent and tell the tracker that we've stopped participating in the swarm. This operation cannot fail." See: "Libtorrent," libtorrent, accessed February 5, 2025, https://libtorrent.org/reference-Session.html#remove-torrent.

[182] META-KADREY-SC-000212. Lines 27-30.

"**is_seed**" function "checks whether a given file has been fully downloaded, at which point the file has become a seed for distribution to other peers in the network."[183] The Krein Report misstates the function of this loop, generally referencing it in his discussion of the torrent downloader scripts,[184] only acknowledging the actual purpose of it at the end of an appendix.[185] **Figure 2** below demonstrates Meta's implementation of the "**is_seed**" loop.

Figure 2 - Sixty Second Polling of is_seed() Loop[186]

[redacted]

99. [redacted]

[183] Krein Report ¶¶127, 168.

[184] Krein Report ¶162.

[185] Krein Report ¶212.

[186] META-KADREY-SC-000213. Lines 85-93.

[187] Although the download_spark.py script uses Python's "logging" library, no filename is specified in the logging.basicConfig, resulting in the logs being written out to the console during download, and not to a persistent file. In any case, these log statements would not provide information on whether particular pieces, or the identity of those pieces, were uploaded from Meta's instances. The logging in download_spark.py is instead limited to reporting download progress over time. META-KADREY-000212.

100. In my opinion, it is highly unlikely that seeding of any of Plaintiffs' works, would have occurred during this short time window after the download of the torrent file was complete. As I will explain in more detail in **Section VI.C.2** below, each of the Plaintiffs' works represents a very small fraction of the content of the downloaded payload, so the likelihood that a peer would have sought to download a piece containing one of Plaintiffs' works, during the small time window between when Meta completed the download and disconnected the torrent, is exceedingly low. Meta also implemented network-level safeguards blocking ingress traffic (**Section VI.C.1** below) that further limit the possibility of seeding Plaintiffs' works to leechers, by accepting connections only from those peers to whom Meta previously initiated a connection and thus rejecting all new incoming requests, including those that might be attempted within the 60 second window after torrent download completion. In sum, Meta took steps to prevent seeding data downloaded via BitTorrent, and these steps should have prevented any distribution of Plaintiffs' works by Meta. As a result, it is not surprising that the Krein Report does not point to any evidence that seeding actually occurred.

C. Seeding Plaintiffs' Works is Also Unlikely Due to Meta's Network Configuration and Other Factors

101. In **Sections VI.C.1** below I discuss how network-level safeguards, the size and scale of the at-issue datasets, the proportion of those datasets that comprise Plaintiffs' works, other libtorrent defaults, and BitTorrent protocol behavior generally, all make it unlikely that Plaintiffs' works were seeded by Meta.

1) Meta's Network Configuration Blocks Inbound Traffic

102. The Krein Report claims that "the download_spark.py script includes a call to the is_seed() function, which checks whether a given file has been fully downloaded, at which point the file has become a seed for distribution to other peers in the network."[188] This statement, at best, describes default functionality of the BitTorrent protocol, and does not take into account the safeguards that Meta implemented to reject incoming requests from other peers on the network. As I explain below, the network environment that Meta used to perform the torrent downloads

[188] Krein Report ¶168.

disallows inbound requests from connections that have not been initiated by Meta, further reducing the possibility of seeding.

103. A ***firewall*** is a network security system that monitors and regulates incoming and outgoing traffic based on predefined security rules. It acts as a barrier between a trusted internal network and untrusted external networks, such as the internet, to prevent unauthorized access and communication.[189]

104. I confirmed with a Meta engineer that the torrent downloads initiated in 2024 took place on an Amazon Web Services (AWS) instance, which is also consistent with contemporaneous documents and deposition testimony provided in the case.[190,191,192] AWS is a cloud services platform that offers a number of services including processing and computing power, storage, database technologies, analytics, and other features.[193] AWS services are often provided through AWS "instances," which are computing resources made available to customers such as Meta.[194] In Meta's AWS cloud environment, torrenting was performed within a Virtual Private Cloud (VPC)—a logically isolated virtual network within a public cloud[195]—where "security groups" act as virtual firewalls. These security groups control traffic at the instance level by specifying rules that determine which connections are allowed or denied.[196]

[189] "What Is a Firewall?," Cisco, accessed February 5, 2025, https://www.cisco.com/site/us/en/learn/topics/security/what-is-a-firewall.html.

[190] "What is an Instance in Cloud Computing?," Amazon, accessed February 8, 2025, https://aws.amazon.com/what-is/cloud-instances/.

[191] Deposition of David Esiobu, December 13, 2024, 168:14-21 (discussing Exhibit 811, Meta_Kadrey_00108336).

[192] Interview with Meta engineers Xiaolan Wang and David Esiobu. Mr. Esiobu confirmed that he set up the AWS instances for the torrent download for Ms. Wang, which Ms. Wang used for the actual download. *See also* Deposition of David Esiobu, December 13, 2024, 168:14-21 (discussing provisioning AWS instances for Ms. Wang to perform torrent download process).

[193] "Cloud Computing with AWS," accessed February 8, 2025, https://aws.amazon.com/what-is-aws/.

[194] "What is an Instance in Cloud Computing?," Amazon, accessed February 8, 2025, https://aws.amazon.com/what-is/cloud-instances/.

[195] "What Is Amazon VPC? - Amazon Virtual Private Cloud," accessed February 5, 2025, https://docs.aws.amazon.com/vpc/latest/userguide/what-is-amazon-vpc.html.

[196] "Control Traffic to Your AWS Resources Using Security Groups - Amazon Virtual Private Cloud," accessed February 5, 2025, https://docs.aws.amazon.com/vpc/latest/userguide/vpc-security-groups.html.

105. In AWS, a software tool referred to as "Terraform" is used to configure security settings, including firewalls in the form of security groups. By default, AWS security groups operate on a "deny-all" model for inbound traffic, blocking any connections that are not explicitly permitted.[197] Updates to security groups can be managed through Terraform configuration files. For example, a security group can be defined using the "aws_security_group" block in a Terraform file, specifying ingress rules to control incoming network traffic.[198]

106. I obtained the Terraform configuration file for the AWS instances used to perform the torrent downloads of Anna's Archive in 2024, which I understand was stored on the Meta source code computer made available during discovery.[199] Meta defined two explicit ingress rules in its Terraform configuration file to allow (i) Secure Shell (SSH) connections and (ii) internal traffic.[200] These two rules modify the default behavior of the security groups, which is to block all inbound requests, effectively defining two narrow 'exceptions' to the deny-all inbound traffic default configuration. The first rule permits SSH traffic specifically to port 22, but only from a predefined set of IP addresses associated with Meta VPNs.[201,202] This ensures that only authorized users connecting through the VPN can securely access the AWS cloud resources via SSH. The second rule allows unrestricted traffic between AWS resources within

[197] "[N]o inbound traffic is allowed until you add inbound rules to the security group" "Security Group Rules - Amazon Virtual Private Cloud," accessed February 5, 2025, https://docs.aws.amazon.com/vpc/latest/userguide/security-group-rules.html.

[198] "Aws_security_group | Resources | Hashicorp/Aws | Terraform | Terraform Registry," accessed February 5, 2025, https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/security_group.html.

[199] [redacted]. During my interview with him, Mr. David Esiobu confirmed that this was the Terraform network configuration file he used for the AWS instances used for the Anna's Archive torrent download that took place in 2024. A copy of this configuration file has been provided with my report.

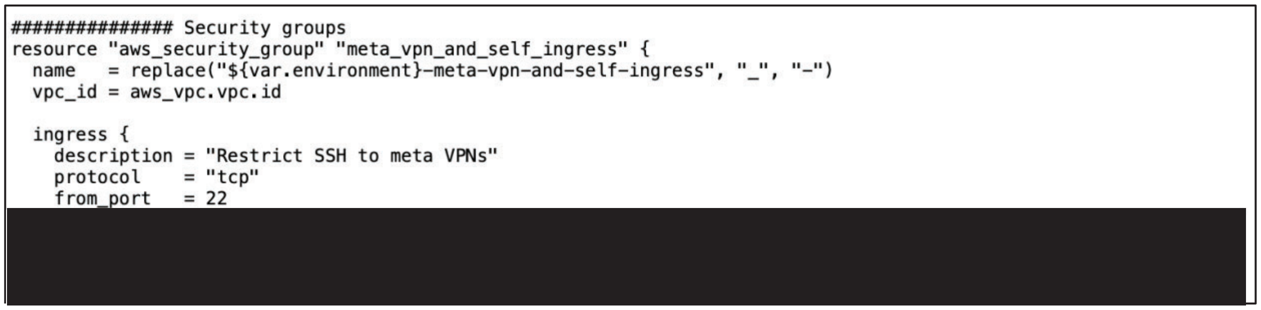
[200] [redacted]

[201] A Virtual Private Network (VPN) is a secure, encrypted connection that allows users to access a private network over the internet, ensuring data privacy and protection from unauthorized access. In cloud environments, VPNs are often used to securely connect remote users or on-premises networks to cloud resources, enabling secure communication as if they were within the same internal network. "What Is a Virtual Private Network (VPN)?," Cisco, accessed February 5, 2025, https://www.cisco.com/c/en/us/products/security/vpn-endpoint-security-clients/what-is-vpn.html.

[202] [redacted]

the same security group.[203] Neither of these exceptions permit leechers to originate new inbound BitTorrent connections.

Figure 3 – Ingress Rule Allowing SSH Only From Meta VPNs[204]



```
############### Security groups
resource "aws_security_group" "meta_vpn_and_self_ingress" {
  name   = replace("${var.environment}-meta-vpn-and-self-ingress", "_", "-")
  vpc_id = aws_vpc.vpc.id

  ingress {
    description = "Restrict SSH to meta VPNs"
    protocol    = "tcp"
    from_port   = 22
```

[redacted]

Figure 4 – Ingress Rule Allowing All Traffic from Instances With the Same Security Group[205]

[redacted]

107. Consequently, Meta's network configuration restricted the circumstances under which Meta could potentially seed data to other peers in a torrent swarm. Because all inbound connections are blocked by default, leechers would have been unable to initiate direct requests for pieces from Meta unless Meta was the originator of the connection. Security groups are stateful, only permitting return traffic for outbound connections, meaning Meta must have first established an outbound connection to a peer before that peer can receive any pieces from Meta.[206] This setup restricts incoming requests for downloading data from Meta and ensures that Meta could only upload a piece of a file to a peer in situations where Meta had

[203] [redacted] Lines 57-63.

[204] [redacted] Lines 49-55.

[205] [redacted] Lines 57-63.

[206] "Security groups are stateful. For example, if you send a request from an instance, the response traffic for that request is allowed to reach the instance regardless of the inbound security group rules." "Control Traffic to Your AWS Resources Using Security Groups - Amazon Virtual Private Cloud," accessed February 5, 2025, https://docs.aws.amazon.com/vpc/latest/userguide/vpc-security-groups.html.

affirmatively initiated the connection to the peer, reducing the likelihood of seeding data to peers in the swarm.

108. To verify that Meta's network configuration functions as intended, I designed a brief experiment using an AWS environment configured identically to Meta's using the Meta Terraform configuration file settings. From my experiment, I confirmed that the AWS instances used by Meta to perform the torrent downloads would have blocked any incoming requests to their IP addresses. My findings and methodologies are described in **Appendix A** below.[207]

109. In sum, the AWS environment and torrenting source code scripts used for Meta's torrenting process provided multiple independent safeguards against seeding with other leechers in the swarm. I will describe below how these, along with other factors, would have made it highly unlikely that Meta would have seeded Plaintiffs' works to other peers on the network.

2) Plaintiffs' Works Constitute Exceedingly Small Portions of the Datasets at Issue

110. The size and scale of the at-issue datasets, relative to the size of the Plaintiffs' works in those datasets, is another important consideration in assessing the likelihood, or unlikelihood, that Meta would have seeded Plaintiffs' works to other peers. Using the data from the comprehensive manual Anna's Archive search that I discussed in **Section V.B.2**, I identified the occurrences of each of the Plaintiffs' works in torrent files for each dataset that was downloaded using the BitTorrent protocol. **Table 2** below summarizes the total number of times Plaintiffs' works were identified in each of the datasets downloaded by Meta, using the BitTorrent protocol:

[207] As I explained in **Section V.B.2**, there is no evidence that the torrent downloads in 2023 of the "scimag" portion of LibGen involved any of Plaintiffs' works. I note, however, that the Meta dev server system used by Mr. Bashlykov to perform those downloaded appeared to have included substantially similar firewall protections as the AWS instances described in the text. *See* 30(b)(6) Deposition of N. Bashlykov, December 6, 2024, 50:9-22 ("Q. Did you look at any other documents that discuss seeding? A. I looked into the internal documentation which specifies which ports open on the dev servers. Q. '… internal documentation which specifies which ports open on the dev servers.' What does that mean? A. So Meta dev servers, they have some firewall protection. I refreshed my recollection on what ports are open to the outside or which ones are not."). I interviewed Mr. Bashlykov, who confirmed that he used Meta dev servers to do the "scimag" torrent downloads in 2023.

Table 2 – Number of Downloaded Plaintiffs' Works Across Torrented Datasets

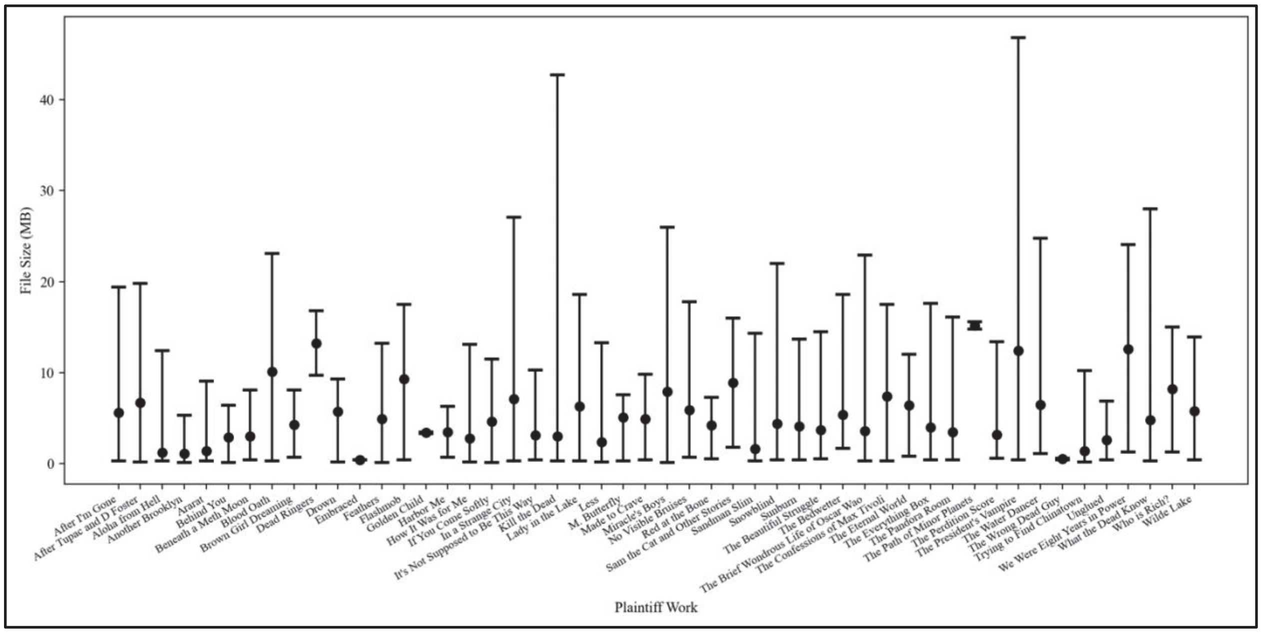
Dataset	Total Number of Instances of Downloaded Plaintiffs' At-Issue Works in Dataset
Libgen.rs Non-Fiction (Scitech Only)[208]	3
Libgen.rs "scimag"	0
Internet Archive (IA)	174
Z-Library (ZLib)	489

111. The first item in **Table 2** shows that, for Libgen.rs Non-Fiction, only three copies of Plaintiffs' works were downloaded. This does not represent three distinct books; it represents a single book (Ta-Nehisi Coates, *The Beautiful Struggle*), which was repeated three times in the downloaded dataset. No other Plaintiff works are contained in this dataset. The repetition of books is a prevalent characteristic of these datasets; as shown by the numbers for IA and ZLib above, Plaintiffs' works often appear multiple times within each dataset.

112. Even taking this duplication into account, the size of each Plaintiff work occurrence is not consistent within or across the at-issue datasets. For example, the file sizes associated with occurrences of some Plaintiff books may be disproportionately larger than other books due to factors such as embedded graphics or the automated text recognition program used for text extraction. Because these factors may impact the proportion of each Plaintiff work within a dataset, they also impact the likelihood that a piece of the work could be seeded to other peers on the BitTorrent network. **Figure 5** below presents an analysis of the wide range of file sizes a given Plaintiff work exhibits across all at-issue datasets. For each work, the vertical bar represents the variation in size of different copies of the work as they occur in the at-issue datasets. See **Appendix B** for details on methodology of identifying the size of the Plaintiffs' works within the at-issue datasets.

[208] Only includes Plaintiffs' works in torrent files downloaded by Meta identified as part of the Scitech portion of Libgen.rs Non-Fiction (r_3650000.torrent to r_ 4142000.torrent). Torrents - Anna's Archive," accessed February 5, 2025, https://web.archive.org/web/20240324090640/https://annas-archive.org/torrents/libgen_rs_non_fic.

Figure 5 – Plaintiff Work File Size Variations Across At-Issue Datasets[209]



113. Regardless of these differences, the proportion of each Plaintiff work (and all Plaintiffs' works combined) to the overall size of each at-issue dataset is negligible. To calculate this proportion, the file sizes were summed across all occurrences of Plaintiffs' works in each dataset, then divided that by the sum of the file sizes downloaded for the overall dataset. Metrics on the combined file sizes and the associated proportion of the Plaintiffs' works are summarized in **Table 3** below. See **Appendix B** for details on methodology of identifying the proportion of the Plaintiffs' works within the at-issue datasets.

[209] Some Plaintiffs' works titles truncated for visual purposes.

Table 3 - Size of Plaintiffs' Works Across Torrented Datasets

Dataset	Total Size of Downloaded Dataset[210,211]	Proportion of Plaintiff Works[212]			
		Average	Lowest	Highest	Total
Libgen.rs Non-Fiction (Scitech Only)	10.3 TB[213]	7.5 MB (0.00007%)[214]			
Internet Archive (IA)	193.5 TB	48.2 MB (0.000024%)	6.8 MB (0.0000033%)	215.9 MB (0.00011%)	2311.9 MB (0.0011%)
Z-Library (ZLib)	63.6 TB	11.0 MB (0.000017%)	0.1 MB (0.00000015%)	57.2 MB (0.000086%)	518.5 MB (0.00078%)

114. Even the highest proportion of all occurrences of a Plaintiff's work over any of the at-issue datasets is a negligible percentage. For example, there are 12 occurrences in IA of the Plaintiff Laura Lippman's work, *What the Dead Know*, but this makes up only 0.00011% of the data in that dataset (with a combined file size of 215.9 MB across all occurrences). This represents the highest proportion of any individual Plaintiff work (by size) for any of the at-issue datasets. **Figure 6** illustrates the vast scale of IA relative to this Plaintiff work.

[210] Process for calculating total download size for each at-issue dataset is discussed in **Appendix B**.

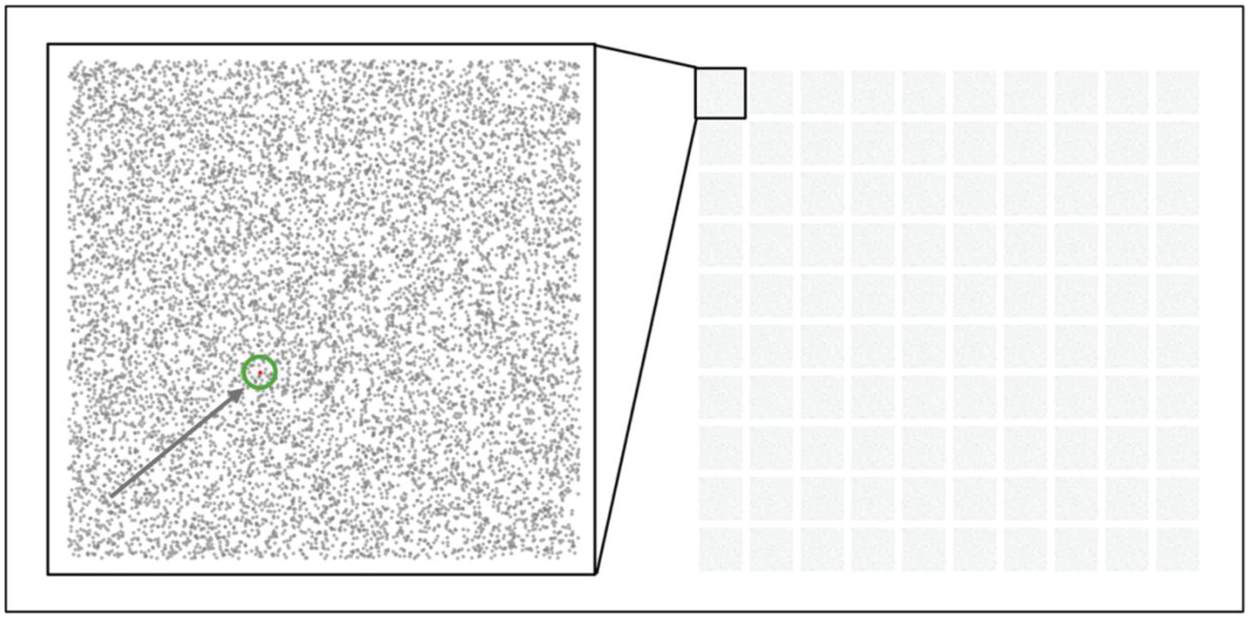
[211] These calculated total download sizes for each of the at-issue datasets are consistent with internal communications in Meta_Kadrey_00107954.

[212] Metrics calculated using aggregated file sizes of all occurrences of a Plaintiff's work in the given dataset.

[213] Only includes Plaintiffs' works in torrent files downloaded by Meta identified as part of the Scitech portion of Libgen.rs Non-Fiction (r_3650000.torrent to r_ 4142000.torrent). Torrents - Anna's Archive," accessed February 5, 2025, https://web.archive.org/web/20240324090640/https://annas-archive.org/torrents/libgen_rs_non_fic.

[214] Only one unique Plaintiff work in Scitech portion of Libgen.rs Non-Fiction, therefore, all metrics are the same when grouped by work.

Figure 6 – Scale of At-Issue Dataset to a Plaintiff Work



115. A single point within a vast collection illustrates the minimal proportion of a Plaintiff's work relative to the datasets that contain them. On the left of **Figure 6**, 10,000 points are displayed, with one highlighted in red. On the right of **Figure 6**, 100 of these 10,000-point plots collectively represent the scale of the entire IA dataset. Only one of these 100 plots contains a single red point, visually depicting the 0.00011% proportion of *What the Dead Know* within the dataset, the highest of any Plaintiffs' works in any of the at-issue datasets.

116. For the ZLib dataset, all of Plaintiffs' books combined make up 0.00078% (78 ten-thousandths of one percent) of the downloaded dataset, as shown in **Table 3**. For example, if one visualizes the ZLib dataset as a book with a million pages, all of Plaintiffs' works combined would make up roughly eight pages of that book. Similarly, if ZLib were represented as the 52 mile distance between San Jose and San Francisco, Plaintiffs' books combined would stretch roughly two feet.[215] For the IA dataset, all of Plaintiffs' books combined make up 0.0011% (11 thousandths of one percent) of the downloaded dataset, as shown in **Table 3**. Using the same analogy of a book with a million pages representing the IA dataset, all of Plaintiffs' works combined would make roughly 11 pages of that book. And

[215] 0.00078% of 274,560 (number of feet from San Jose to San Francisco, based on 52 miles * 5280 feet per mile).

using the same analogy of the distance from San Jose to San Francisco, all of Plaintiffs' books in the IA dataset would stretch roughly three feet.

117. The proportion of Plaintiffs' works to the sizes of the torrent files that contain them was also calculated. First, the maximum number of pieces that each work occurrence could span was calculated by dividing the work occurrence file size by the piece size, rounding up and adding one.[216] These values were summed to calculate the total number of pieces across all torrent files in which Plaintiffs' works appear. Metrics on the number of torrents downloaded by Meta that contained Plaintiffs' works, and the proportion of Plaintiffs' works to other data in the torrents, is illustrated in **Table 4** below. See **Appendix B** for details on methodology of identifying the proportion of the Plaintiffs' works within the at-issue torrents.

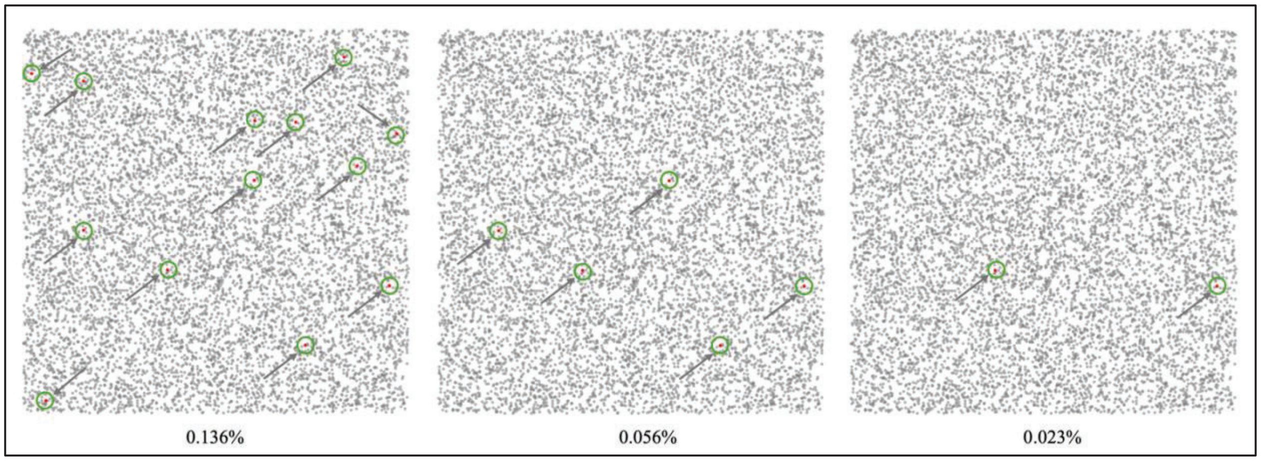
Table 4 - Proportion of Plaintiffs' Works Across Downloaded Torrent Files

Dataset	Total Torrents Downloaded	Total Number of Torrent Files Downloaded Containing Plaintiffs' Works	Proportion of Pieces in Torrent Files Containing Plaintiffs' Works
Libgen.rs Non-Fiction (Scitech Only)	491	2	0.136%
Internet Archive (IA)	143	46	0.056%
Z-Library (ZLib)	265	146	0.023%

118. The proportions of all occurrences of a Plaintiff's work over any of the at-issue torrents is a negligible percentage. For example, there are 46 torrent files with at least one of the Plaintiffs' works in IA, however, only 0.056% of the pieces in these 46 torrent files contain any part of the Plaintiffs' works. **Figure 7** illustrates the vast scale of all the pieces in the IA torrents containing Plaintiffs' works relative to pieces that actually contain Plaintiffs' works.

[216] Calculating the maximum number of pieces containing a given work occurrence using this method is an overestimate and assumes the worst case: that each of the Plaintiffs' works is spread out across the maximum number of pieces that it could be based on the file size and piece size for the torrent. Therefore, the proportions provided in **Table 4** are all upper-bound values.

Figure 7 – Visualization of Proportion of Plaintiffs' Works Within Libgen.rs Non-Fiction, Internet Archive, and Z-Library Torrent Files that Contain Plaintiffs' Works



119. Each box in **Figure 7** contains 10,000 points, as a representation of the torrents downloaded by Meta within Libgen.rs Non-Fiction (Scitech only), IA, and ZLib. The red dots represent the proportion of Plaintiffs' works pieces within the torrent files that contain any Plaintiffs' works: 13 out of 10,000 for Libgen.rs Non-Fiction, 5 out 10,000 for IA and 2 out of 10,000 for Z-Lib. These dots visually represent the small percentages of Plaintiffs' works within the torrent files that contain them, that were downloaded by Meta.

120. For the ZLib dataset, all the pieces containing Plaintiffs' works combined make up 0.023% of the torrent files that contain Plaintiff works (a subset of all downloaded torrent files), as shown in **Table 4**. If ZLib were represented as the 52 mile distance between San Jose and San Francisco, as analogized above, all the pieces of the Plaintiffs' works would be a little less than two bus lengths.[217] For the IA dataset, all the pieces of the Plaintiffs' works combined make up 0.056% of the torrent files that contain Plaintiff works. Using the same analogy of the distance from San Jose to San Francisco, all of Plaintiffs' books in the IA dataset would stretch roughly over four bus lengths.

121. The small proportion of Plaintiffs' works to the sizes of the at-issue datasets, as well as the small proportion of Plaintiffs' work pieces within the torrents they exist in, are important factors that further drive down the possibility that these works would have been seeded to other

[217] 0.023% of 274,560 (number of feet from San Jose to San Francisco, based on 52 miles * 5280 feet per mile). One bus length is roughly 35 feet.

peers on the network. Leechers do not download an entire payload from a single seeder when multiple seeders are present in the swarm. For Meta to have seeded a piece containing one of Plaintiffs' works to a leecher, therefore, would have required: that (i) Meta had previously initiated a connection to the leecher and that connection was still open (as any leecher to whom Meta did not initiate a connection would have been blocked by the firewall rules as described above), (ii) Meta has used one of its unchoke slots to unchoke that leecher, rather than some other peer, and then (iii) the leecher asked Meta (rather than some other peer to which the leecher was connected) for that particular piece (out of the vast number of other pieces that do not contain any of Plaintiffs' works). Because Plaintiffs' works individually and collectively constitute a negligible percentage of the overall payload, it is unlikely that Meta was asked to seed the precise pieces containing Plaintiffs' works. This is further reinforced by (iv), the 60 second period described above, which provides the maximum window in which any seeding could have theoretically occurred, after which the torrent file is removed immediately. The additional constraints imposed by libtorrent settings and the BitTorrent protocol, which I discuss in more detail in the next section, make this even less likely to have occurred.

3) Libtorrent Default Settings and BitTorrent Protocol Behavior Further Diminished the Likelihood of Seeding

122. The Krein Report does not consider important default settings in the libtorrent library used by Meta,[218] and other aspects of the BitTorrent Protocol that further limit Meta's ability to seed. As previously discussed in **Section IV.B**, the choking and unchoking process determines the number of peers with whom a leecher can exchange data at any given time. Seeding is restricted by the applied choking algorithm, which can limit the number of peers it can upload to at once using a pre-determined number of unchoke slots.

123. The default libtorrent choking algorithm used during seeding limits the number of unchoke slots. During seeding, libtorrent uses the **round_robin** algorithm, which maintains a constant maximum of unchoked slots, but periodically round-robins the peers that are unchoked while seeding. The **unchoke_slots_limit** setting determines the maximum number of concurrently unchoked peers. Further, the **unchoke_interval** determines how often peers are re-evaluated

[218] Interview with Meta engineer, Xiaolan Wang. I confirmed that Meta did not modify the libtorrent source code.

for being choked/unchoked. Using the same libtorrent version that Meta used for the torrent download,[219] I verified the default settings to confirm the choking algorithm, maximum number of unchoke slots, and the unchoke interval. As a result, Meta could not have been uploading data to more than eight peers at any specific point in time while seeding, and even then, any peers within these eight could have been rotated for a choked peer, every 15 seconds.[220] Coupled with the extremely small percentage of pieces in the at-issue torrent payloads that comprise Plaintiffs' works, this further limits the likelihood that Plaintiffs' works would have been seeded by Meta to other peers.

124. Further, the mere presence of leechers in a torrent swarm does not necessarily indicate the seeding of Plaintiffs' works by Meta. At any given time, the number of leechers in the swarm fluctuates, and not all leechers require every piece of the torrent. Leechers that are downloading torrents over time may have already obtained the pieces containing Plaintiffs' works from other peers on the network or in prior paused sessions, eliminating the need for those leechers to obtain those pieces from Meta. Even if a leecher in the swarm is seeking a piece that contains one of Plaintiff's works, Meta could only provide the piece if it was currently connected and had currently unchoked the leecher, and if the leecher chose to send that particular piece request to Meta as opposed to another seeder or leecher that had the piece. The pieces prioritized for download by other leechers will also change dynamically.

125. The rarest-piece-first strategy in the BitTorrent protocol, which I discussed in **Section IV.C.2**, prioritizes downloading the least common pieces of a torrent first. Because the rarity of pieces changes dynamically based on the composition of the swarm, and the availability within the swarm of specific pieces at any given moment, it is not possible to determine which pieces were considered "rare" when Meta was downloading the at-issue torrents in 2024, and given the constantly shifting nature of piece rarity, it is impossible to determine with certainty whether pieces containing Plaintiffs' works were ever prioritized as "rare" or deprioritized.

[219] Version 2.0.9, based on an interview with Xiaolan Wang. I also confirmed that version 2.0.9 would have been the latest stable release of libtorrent at the time of torrent download in or about April 2024. *See,* "Libtorrent: Python Bindings for Libtorrent-Rasterbar," accessed February 10, 2025, https://pypi.org/project/libtorrent/2.0.9/#history.

[220] There is also a concept of optimistic unchoking, wherein libtorrent reserves 20% of the unchoke slots (1, given 8 as default), to unchoke a new choked peer every 30 seconds.

Nevertheless, even if pieces containing Plaintiffs' work were rare at one point in time, all of the seeders in the swarm would be known to have had them, and other leechers already in the swarm who did not already have the piece would have been prioritizing downloading the rarer pieces from other peers even before Meta joined. On the other hand, if Plaintiffs' works were already well-distributed among the swarm, it is even less likely that Meta would have seeded those pieces to other peers, given their low rarity.

4) Likelihood that All Necessary Conditions Aligned for Meta to Upload Plaintiff's Works During Download is Substantially Limited

126. As I outline above, several factors had to have aligned sequentially and simultaneously for Meta to have seeded pieces of a torrent that contain Plaintiffs' works to leechers:

127. **Meta must have previously initiated a connection with the leecher.** This would have been required in light of the firewall protections described in **Section VI.C.1**. This would also have been unlikely to have occurred during the 60 second maximum time window in which seeding could have theoretically occurred because, at that point, Meta would not have required pieces of that torrent file from other peers and thus would not have initiated connections with other peers.

128. **The leecher must not have already possessed the pieces containing Plaintiffs' works.** Obviously, the leecher may not have required all, or any, of the particular pieces containing Plaintiffs' works (which as explained comprise a tiny fraction of the at-issue datasets). Leechers that existed in the swarm and who had been actively downloading may already have possessed the pieces of the torrent containing Plaintiffs' works.

129. **The leecher must have been requesting the specific pieces containing the small portions of Plaintiffs' works.** Even if Meta had connected to the leecher, and the leecher had not already downloaded the pieces containing the Plaintiffs' works, the next condition would have been that the leecher must have been actively requesting one of those pieces (either because the piece was randomly selected or because the piece was deemed rare). Although piece rarity is dynamic and cannot be determined for the time Meta conducted the download, this condition would also need to have been aligned (except on a peer's initial requests, when a small number of pieces may be requested randomly). In any case, it must have been aligned

that the pieces requested by the leecher were the ones that contained Plaintiffs' works. As noted in **Section VI.C.2**, this is a rare occurrence given the negligible percentage of pieces in the at-issue datasets that contained Plaintiffs' works.

130. **The leecher must select Meta as the seeder to upload these pieces.** At this stage, the leecher must have also decided to select Meta as the peer from which to download the pieces and sent its piece request to Meta. Therefore, for Meta to have seeded pieces containing Plaintiff's works, two additional key factors must have aligned: (i) the sequential satisfaction of all prior conditions that make Meta a viable source for the piece, and (ii) the simultaneous, selection of Meta as the preferred peer for the download over all other available peers that have the piece. Only if both of these additional conditions were met could Meta have been the particular peer that seeded Plaintiffs' works to leechers.

131. **Meta must have been able to seed pieces containing Plaintiffs' works within a maximum of 60 seconds.** As outlined in **Section VI.B.2**, Meta's torrent download scripts limited the amount of time a torrent could theoretically have been seeding. By polling the **h.is_seed()** loop every 60 seconds, Meta set an upper bound for how long a torrent could have theoretically been seeded after torrent download completion, as the file is removed as soon as **h.is_seed()** is returned as true. Therefore, for Meta to have seeded pieces containing Plaintiffs' works, all the prior conditions need to have been sequentially satisfied, Meta had to have been chosen as the peer from which to download, and the upload of the pieces to the leecher must have occurred within this brief time window before the torrent was removed.

132. **A leecher must have remained in Meta's unchoked slots long enough to download a piece containing Plaintiffs' works.** As discussed in **Section VI.C.3**, during seeding, libtorrent uses a round-robin method of unchoking, checking the eight unchoke slots every 15 seconds, and potentially rotating out peers. Further, optimistic unchoking rotates out 1 unchoked leecher for a choked leecher every 30 seconds. Therefore, even though Meta could potentially have been seeding for a maximum of 60 seconds after completion of the torrent download, a leecher would have had to occupy one of the eight unchoked slots, with a possibility of only being allowed to download for 15 seconds before being choked again.

133. To recap, for Meta to have seeded pieces containing Plaintiffs' works, all of the following conditions must have coalesced: (i) Meta must have previously and affirmatively initiated a connection with a leecher, (ii) the leecher must not have already downloaded the pieces containing Plaintiffs' works and must also be prioritizing those pieces for download, (iii) the pieces being prioritized must contain the portion of data out of all the data in the dataset that contains a given Plaintiff's works, (iv) out of all the peers the leecher is connected to that also have the piece, Meta must have been the one chosen as the source for seeding, (v) Meta must seed pieces to the leecher that contain Plaintiff works' within the (at maximum) 60 seconds of Meta's torrent download completion, and finally (vi) a leecher must remain in Meta's unchoked slots long enough to download a piece containing Plaintiffs' works.

134. All of these constraints and conditions must have been met in order for Meta to have seeded a piece containing the Plaintiffs' works. Considering the factors discussed above, any contention in the Krein Report that Meta uploaded Plaintiffs' works to peer on the BitTorrent network (which is not demonstrated with any evidence) would be nothing more than speculation and conjecture, given the many factors identified above. In light of all of the factors discussed above, in my opinion, it is exceedingly unlikely that Meta seeded any of the Plaintiffs' works.

VII. CONCLUSION

135. In this report, I respond to several inaccurate, misleading, and incomplete assertions the Krein Report makes regarding Meta's use of "BitTorrent," the potential "seeding" of data, and Meta's environment for performing torrent downloads. Based on my analysis, I demonstrate that the Krein Report (i) misunderstands and mischaracterizes the BitTorrent protocol when suggesting Meta necessarily seeded data, (ii) does not consider the safeguards implemented by Meta to prevent the seeding of downloaded data nor the myriad of unlikely factors that would have needed occur for seeding to take place, and (iii) presents no evidence that Plaintiffs' works were actually seeded by Meta (as opposed to other data in the at-issue datasets, which has also not been demonstrated), and rather only speculates that this was the case based on the existence of torrenting scripts in Meta's codebase.

136. As I discuss in **Section VI.B** and **Section VI.C**, Meta had safeguards built in that prevent seeding such as (i) removing the torrent from the session within no more than 60 seconds after the torrent download has completed, and (ii) blocking all unsolicited inbound connections, effectively rejecting requests from un-connected leechers during the brief 60 second period.

137. Further, in order for Meta to have seeded Plaintiffs' works, many different practical and technical factors would have had to come together that, in my opinion, make it highly unlikely that this occurred, as described in the preceding section. I note that the Krein Report does not present any evidence that Meta actually seeded data, let alone that Meta seeded any of the Plaintiffs' works.

138. In sum, based on the safeguards implemented by Meta, the numerous factors in the torrent network that would have needed to align, and the small proportion of Plaintiffs' works in the at-issue datasets (as discussed in **Section VI.C.2**), it is highly unlikely that Meta seeded the Plaintiffs' works to other peers during any of the torrent downloads discussed above.